NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

AERODYNAMIC ANALYSIS OF A MODIFIED, PYLON-MOUNTED JSOW/CATM USING MULTI-GRID CFD METHODS

by
Boaz Pomerantz

March, 1997

Thesis Advisor: Oscar Biblarz
Thesis Co-Advisor: Garth Hobson

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 1997	Engineer's Thesis

4. TITLE AND SUBTITLE
AERODYNAMIC ANALYSIS OF A MODIFIED, PYLON-MOUNTED JSOW/CATM USING MULTI-GRID CFD METHODS

5. FUNDING NUMBERS

6. AUTHOR(S)
Boaz Pomerantz

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

Computational Fluid Dynamics (CFD) has become a major tool in aerodynamic analysis throughout the aerospace industries, complementary to traditional methods such as wind tunnel testing, and analytical calculations. In this research, an attempt was made to integrate the Similarity and Area Rules with CFD methods. Both tools, the Similarity/Area-Rule and CFD are used to derive the characteristics of complicated aerodynamic shapes in the transonic Mach number regime. It was found that the Similarity Rule can only be verified qualitatively. On the other hand, the Area Rule can be more completely verified. The aim was to find ways to minimize the drag of the training configurations of the Air-to-Ground (A/G) weapon, Joint-Standoff-Weapon (JSOW), in its Captive-Air-Training-Missile (CATM) configuration. By analyzing the combination of CATM and Pylon, it was found that the drag of this configuration depends on the average slope of the area cross-section distribution of the afterbody. The CFD tools used were a state-of-the-art grid generation code, GRIDGEN, and a multi-grid integration code, PEGSUS; the configurations were run with the OVERFLOW solver using Euler, as well as Navier-Stokes solutions. For drag optimization, Euler solutions give adequate results, the need for NS solution can be restricted to more intensity viscous analysis.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Computational Fluid Dynamics, Area Rule, Similarity Rules, Transonic flow, JSOW, CATM	169
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

Approved for public release; distribution is unlimited

AERODYNAMIC ANALYSIS OF A MODIFIED, PYLON-MOUNTED JSOW/CATM USING MULTI-GRID CFD METHODS

Boaz Pomerantz
Major, Israeli Air Force
BS Aeronautical Engineering., Technion, Israel, 1986

Submitted in partial fulfillment of the
requirements for the degree of

AERONAUTICAL AND ASTRONAUTICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL
March 1997

Author: _____
Boaz Pomerantz

Approved by: _____
Oscar Biblarz, Thesis Advisor

_____
Garth Hobson, Thesis Co-Advisor

_____
Daniel J. Collins, Chairman
Department of Aeronautics and Astronautics

ABSTRACT

Computational Fluid Dynamics (CFD) has become a major tool in aerodynamic analysis throughout the aerospace industries, complementary to traditional methods such as wind tunnel testing, and analytical calculations. In this research, an attempt was made to integrate the Similarity and Area Rules with CFD methods. Both tools, the Similarity/Area-Rule and CFD are used to derive the characteristics of complicated aerodynamic shapes in the transonic Mach number regime. It was found that the Similarity Rule can only be verified qualitatively. On the other hand, the Area Rule can be more completely verified. The aim was to find ways to minimize the drag of the training configurations of the Air-to-Ground (A/G) weapon, Joint-Standoff-Weapon (JSOW), in its Captive-Air-Training-Missile (CATM) configuration. By analyzing the combination of CATM and Pylon, it was found that the drag of this configuration depends on the average slope of the area cross-section distribution of the afterbody. The CFD tools used were a state-of-the-art grid generation code, GRIDGEN, and a multi-grid integration code, PEGSUS; the configurations were run with the OVERFLOW solver using Euler, as well as Navier-Stokes solutions. For drag optimization, Euler solutions give adequate results, the need for NS solution can be restricted to more intensity viscous analysis.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

The success of this research could not have been achieved without the support of my thesis advisor, Prof. Biblarz. His professional guidance as well as his scholarly attitude made this project one of my most enriching experiences of my academic and professional life. The project became a partnership, and the team effort brought results.

Also, I would like to thank Prof. Hobson for his dedicated attitude and professional help with all the CFD process in the thesis. As an instructor, as well as a friend, he was willing to help, assist, and share his experience and knowledge as if it were his own project.

My studies at the NPS could not have been possible without the support of the Israeli Air Force and my commanders. My contribution to the organization will be significantly enhanced by the experience and knowledge I gained by doing this thesis.

Finally I would like to thank my wife, Ephrat, for her patience in the long days and nights I spent to complete this thesis. Her constant support enabled me to be completely dedicated to the project and to achieve my goals.

I. INTRODUCTION

The development of any new aerodynamic configuration relies heavily on Computational Fluid Dynamics (CFD) to enable an economical study of parameter sensitivities and configuration integration. The Similarity rule, a more traditional technique, was commonly used, prior to CFD, in conjunction with customary methods, such as wind tunnel testing and full scale tests. Similarity rule can be applied to cases under investigation together with other known information about the test object.

As part of a design team that was established at the Naval Postgraduate School (NPS), the Joint Standoff Weapon (JSOW) missile that is under development is being analyzed and geometrically modified to optimize its weight and drag characteristics. The aerodynamic results can then be used as background for studying and verifying the Area-Rule defined in the 1950's by Whitcomb [Ref. 1].

The CFD analysis may be applied using both single-grid and multi-grid techniques. State-of-the-art codes (such as GRIDGEN Ver. 9.6), can be used to create grids, merge them together for physical computational integration and create interpolation regimes, while solving flow around the given configuration both in inviscid (Euler) and viscous (Navier-Stokes) flows by using CFD codes such as NASA's OVERFLOW flow solver. The multi-grid method allows complicated geometrical configurations to be analyzed; however, it is well known that generating grids around configurations that appear simple initially, may turn out to be a complicated process.

The CFD process, apart from involving many primary and secondary codes, is known to be very sensitive to the grid generation process and architecture. OVERFLOW, for example, is sensitive to the distribution of grid lines. Other sensitivities are found in merging codes such as PEGSUS if the interpolation regime boundaries are set improperly, resulting in non-overlapped regimes or "orphan" grid points.

Only after the CFD process is successfully implemented may the Area-Rule study be investigated. This rule, in essence, states that the drag of a body depends (above all) on its cross-area distribution along its axis. Pressure related drag decreases as the distribution of the cross-section area becomes smoother.

A combination of Captive Air Training Missile (CATM) and pylon can be used to investigate the Area-Rule by running different cases for different CATM-pylon relative locations. Methods for achieving area distribution change must be consistent with their ease of implementation with the CFD tools, e.g., modeling the area distribution changes by grid modification is difficult to implement with CFD; hence, using two grids in different axial relative location is much easier.

II. BACKGROUND

A. JSOW WEAPON SYSTEM

The next generation of air-to-ground (A/G) weapon systems, which are under development throughout the world, is a stand-off type of weapon that enables the launcher crew to be out of the threatened area, preferably out of ground-to-air missile range, while still being able to launch and control the weapon.

Several such systems are in various stages of development such as SLAM (Stand-off Land Attack Missile) developed by McDonnell Douglas (MD) for the U.S. Navy. The modified and improved version of the SLAM (SLAM-ER). CASOM (Conventional Armed Standoff Missile) developed by the British. JDAM and the AGM 154 (Air-to-Ground Missile)-JSOW developed under a joint U.S. Navy/Air Force contract by Texas Instruments (TI) as the prime contractor.

Three versions of the operational AGM 154 weapon system are being developed [Ref. 2].

- The basic configuration (AGM-154A) cluster weapon contains 145 Aerojet/Olin BLU-97 submunition (combined effects of shaped charge, blast/fragmentation and incendiary) which is designed to replace the anti-soft armor Rockeye A/G weapon.

- The advanced configuration (AGM-154B) cluster weapon contains 6 Textron Systems BLU-108. Each contains 4 "Skeets" self-homing, anti-armor submunitions.

- The unitary configuration (AGM-154C) blast/fragmentation warhead with a terminal guidance system.

An advanced version will include jet engines to increase the range of the weapon up to 100 nautical miles.

Fulghum [Ref. 3] described the weapon as a "truck" that will be able to deliver its payload to a nominal range of 40 miles and will be able to be carried by most NATO and US aircraft. The future configuration will include smart seekers for the end game. The main candidates for the seekers are SAR, Imaging Infrared, Laser Radar, and millimeter wave radar [Ref. 3]. The weapon's operational principal includes the release from the airframe platform, gliding to its target using a combination of Global Positioning System (GPS)/inertial navigation systems, providing a navigation accuracy of up to 10 m, and dispersing the submunitions or hitting the target using the "man in the loop" preprogrammed schedule. The development of the weapon system itself is, in general, widely embedded with modeling and simulations in order to decrease the evaluation tests [Ref. 4].

B. JSOW TRAINING POD - CATM AND NAVAL POSTGRADUATE SCHOOL TEAM INVOLVEMENT

While the JSOW weapon system is being developed by TI, the Naval Postgraduate School (NPS) as a Navy organization has been cooperating with the contractor to develop the training configuration of the system.

The JSOW training program, in addition to having a sophisticated and expensive systems such as the SLAM and GBU-15, has degenerated configurations that are being used for training purposes only. The CATM configuration is a pod that hangs on the

regular aircraft pylon, simulating the flight condition of the operational version, and has characteristics similar to the operational vehicles.

A working group consisting of professors and students from the Aeronautical and Astronautical Department at NPS was established in 1995 to develop a training pod including aspects such as:

- Software simulation of the JSOW training mission.
- Static and dynamic load analysis.
- Weight optimization (to achieve the 300 lb limit that obviates the need for emergency separation).
- Quantity in fleet optimization for the training pod.
- Aerodynamic analysis for the pod.
- Drag optimization.

The group started a preliminary development of a training pod that would achieve the 300 lb limit and have a relative low drag, thus, enabling longer training missions.

This thesis concentrates on the aerodynamic analysis aspect of the training pod, particularly on the drag optimization of the CATM.

C. CFD - COMPUTATIONAL FLUID DYNAMICS AS A DEVELOPMENT TOOL USED IN THE PROCESS

1. Overview

CFD is a powerful tool for aerodynamic analysis. Its use has accelerated significantly during the last few years as a result of the fast development of supercomputers, which produce solutions within a reasonable time and effort. This by the

solution to an enormous number of simultaneous differential equations which are needed to analyze flow around complicated vehicles.

2. Competitive Methods

Competitive methods for aerodynamic analysis such as wind tunnels, pure analytical and parametric applications, and flight tests suffer from high cost limitations, inaccuracies, and availability difficulties. Ref. 5 quotes the following:

> The Navy requires a predicted method capable of predicting the store's pitching and yawing moments to within 10% of flight data in the transonic flight regime.

Assuming CFD is accurate in store's pitching and yawing moments calculations to within 10% of flight data in the transonic flight regime, it should satisfy the requirement above.

D. THEORETICAL BACKGROUND

1. General

OVERFLOW, the flow solver, developed by NASA Ames was used in this research, it is capable of solving the Euler as well as with Navier-Stokes (NS) equations. These are the inviscid/viscous representations of flow characteristics for any configuration defined by initial and boundary conditions. The detailed characteristics of the code will be discussed later. There are no closed solutions for Euler/Navier-Stokes equations. The analytical closed solutions are restricted to the potential flow theory, e.g., the solution of a flow past a sphere or a flow past a cylinder.

In essence, OVERFLOW is oriented to solve fully viscous flows. The Euler solution is a degenerated form of the full solution achieved by turning off some options

from the full NS solution. The viscous terms, partially defined by the shear expressions are set to zero in an Euler case.

The code is an implicit, conservative solver that can use the Chimera overlapped grid scheme in conjunction with PEGSUS.

2. Navier-Stokes (NS) Equations And Solution Techniques

The NS equations include the viscous effects concentrated mainly in the boundary layer [Ref. 6]. Thus we can introduce the NS equations in their Cartesian form:

Introducing the following variables:

- u, v, w - the flow velocities in the x, y, z directions respectively.
- ρ – the flow density.
- p - the pressure.
- τ - the shear stress tensor.
- e - the internal energy.
- E_t - the total energy.
- q - the heat flux vector.
- k - thermal diffusivity of the fluid.
- μ - viscosity coefficient.
- T - temperature.
- R - universal gas constant.
- C_v - specific heat at constant volume.

$$\frac{\partial Q}{\partial t}+\frac{\partial F}{\partial x}+\frac{\partial G}{\partial y}+\frac{\partial H}{\partial z}=0 \qquad (1)$$

Where Q, F, G, H are defined as follows:

$$Q = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ E_t \end{pmatrix} \qquad (2)$$

$$F = \begin{pmatrix} \rho u \\ \rho u^2 + p - \tau_{xx} \\ \rho uv - \tau_{xy} \\ \rho uw - \tau_{xz} \\ (E_t + p)u - u\tau_{xx} - v\tau_{xy} - w\tau_{xz} + q_x \end{pmatrix} \qquad (3)$$

$$G = \begin{pmatrix} \rho v \\ \rho uv - \tau_{xy} \\ \rho v^2 + p - \tau_{yy} \\ \rho vw - \tau_{yz} \\ (E_t + p)v - u\tau_{xy} - v\tau_{yy} - w\tau_{yz} + q_y \end{pmatrix} \qquad (4)$$

$$H = \begin{pmatrix} \rho w \\ \rho uw - \tau_{xz} \\ \rho vw - \tau_{yz} \\ \rho w^2 + p - \tau_{zz} \\ (E_t + p)w - u\tau_{xz} - v\tau_{yz} - w\tau_{zz} + q_z \end{pmatrix} \qquad (5)$$

Where the shear and heat flux components are defined as:

$$\tau_{xx} = \frac{2}{3}\mu\left(2\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y} - \frac{\partial w}{\partial z}\right) \qquad (6)$$

$$\tau_{xy} = \tau_{yx} = \mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) \quad (7)$$

$$q_x = -k\frac{\partial T}{\partial x} \quad (8)$$

$$\tau_{yy} = \frac{2}{3}\mu\left(2\frac{\partial v}{\partial y} - \frac{\partial u}{\partial x} - \frac{\partial w}{\partial z}\right) \quad (9)$$

$$\tau_{xz} = \tau_{zx} = \mu\left(\frac{\partial w}{\partial x} + \frac{\partial u}{\partial z}\right) \quad (10)$$

$$q_y = -k\frac{\partial T}{\partial y} \quad (11)$$

$$\tau_{zz} = \frac{2}{3}\mu\left(2\frac{\partial w}{\partial z} - \frac{\partial v}{\partial y} - \frac{\partial u}{\partial x}\right) \quad (12)$$

$$\tau_{yz} = \tau_{zy} = \mu\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right) \quad (13)$$

$$q_z = -k\frac{\partial T}{\partial z} \quad (14)$$

And the equivalent equations of state, internal energy and total energy are:

$$p = \rho RT \quad (15)$$

$$e = C_V T \quad (16)$$

$$E_t = \rho\left[e + \frac{1}{2}(u^2 + v^2 + w^2)\right] \quad (17)$$

III. COMPUTATIONAL TOOLS AND TECHNIQUES

A. ARCHITECTURE

One of the main characteristics of CFD work is the complexity of various codes involved in the analysis. It was discovered that certain aspects of CFD, especially grid generation, tend to require more "art" than science to achieve reasonable results.

In this section we will describe the codes that were used in the process and the relationship between them. The techniques for using them is described in detail in Appendix C and Appendix J.

List of codes (not necessarily in the order of use):

1. GRIDGEN - Grid generation program

2. PEGSUS - Determine the interpolation stencil (IBLANK file) between multiple overlapping grids.

3. OVERFLOW - Flow solver program.

4. FAST - Visualization program of flow solution over the investigated blocks.

5. GRIDED - Editing grids program.

6. RMG2PEG - Merges the separated grids into one input file for PEGSUS.

7. MERGE41 - Merges the output files of PEGSUS and IBLANK file into the "grid.in" input file for OVERFLOW.

8. xa2b.f - ASCII to Binary and "xb2a.f", Binary to ASCII transform files.

9. readfomo.m and readresid.m - Matlab files (written by Bret S. Barton [Ref. 7] and modified in this research) makes it possible to present graphically the OVERFLOW output data given in "fomo.out" and "resid.out", respectively.

10. wrq - a conversion file that converts the solution file "q.save" from its Cray format to a SGI format.

11. wrgr - a conversion file that converts the grid file - "grid.in" in its Cray format to a SGI format.

B. GRIDGEN

1. General

The basic element of the CFD analysis is modeling the body under investigation in its environment. Currently, NPS uses the following programs to create meshes:

a. GRIDGEN
b. HYPGEN
c. GRAPE

The code that was used in this research was GRIDGEN, developed by John P. Steinbrenner and John R. Chawner, MDA Engineering, Inc. [Ref. 8]. Version 9.6 of the code was used throughout this research. Currently, version No. 11 is under development. The original version of the code was developed by the U.S. Air Force. Since then, the code became commercial, and attempts by NPS to update that version (instead of the current 9.6 version, which contains some bugs) have not been successful. The code was developed using the Silicon Graphics IrisGL graphics library [Ref. 8], and hence can be used by the computers in the computer laboratory of the NPS Aeronautics and Astronautics Department. The code is menu driven, very user friendly and offers a very convenient way to create an optimal grid around complicated surfaces. Difficulties that were found in the code will be elaborated further on.

2. Theory

In essence, the purpose of the code is to create a block of a large amount of volumetric cells that fill the investigated domain.

Two domains considered in the grid generation and flow solver processes:

- Physical domain, in the grid generation program.
- Computational domain, in the flow solver program.

The physical domain (assigned by x, y, z coordinates) consists of the actual surfaces that can be modeled geometrically. The computational domain (ξ,η,ζ) is a virtual domain of simple cell boxes that can be used easily by the flow solver [Fig. 1].

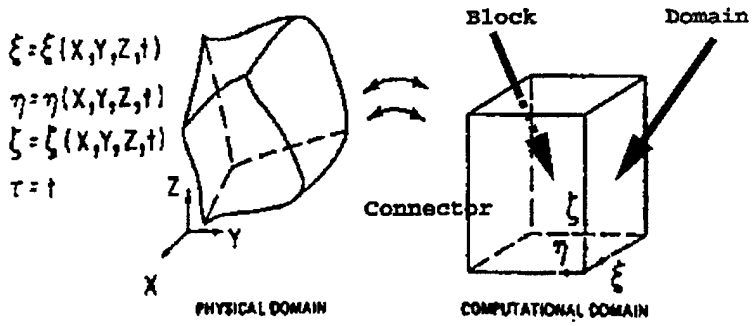


Figure 1. Physical and computational domains used by GRIDGEN [Ref. 6].

The output of the grid generation process is a file that contains all the coordinates of the intersection points of the grid lines.

A grid (or mesh, as it is called in other sources) is a block made of six domains (as a box). The domains are made of four connectors. The basic element is a connector, that is, a line that can form any desired shape and is divided into a certain number of grid points. A connector [Fig. 1] can be made of single or multiple segments. This architecture provides a convenient way to model complicated shapes by dividing the shape contour into many short, accurate segments. The connector is the subelement of the next grid element, the domain [Fig. 1]. The domain is a surface made of 4 connectors: 2 pairs that create a 3D mesh of the surface. In order to achieve a reasonable domain, each two opposite connectors must be the same dimensions (but not necessarily the same grid distribution). In the domain menu options [Fig. 2], one can identify the power of GRIDGEN to create complicated, physical domains, e.g., by default, GRIDGEN provides algebric solver to smooth the grids; however, in complicated configurations, the feature of the Elliptical Solver makes it possible to curve grid lines in the appropriate direction and create a uniform mesh, thus preventing negative cells in OVERFLOW (which will be elaborated in the following sections).

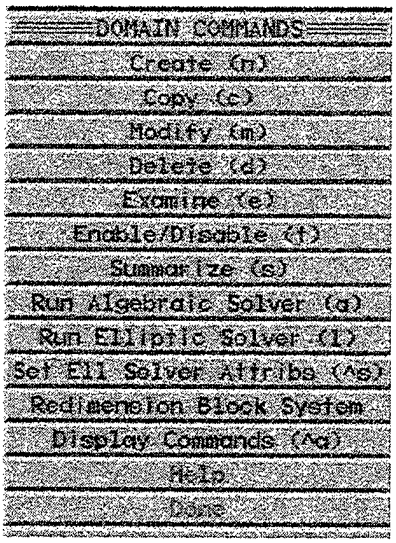


Figure 2. Domain main menu in GRIDGEN.

The domain is the subelement of the upper level element of GRIDGEN, the block. The block is in essence a box made of six facets: the six domains. To achieve a valid block, the dimension of each two opposite domains has to be identical [Fig.1]. GRIDGEN makes it possible to create a block with more than six domains so that the final configuration after summation (joining) of the appropriate domains will be valid, i.e., the grid surface dimensions of each two opposite domains will be the same. In all the blocks that were created in this research, the configuration included one domain that was the investigated domain (i.e., the surface of the CATM or the surface of the pylon, or the surface of the wing became one domain in each of their respective blocks). The other five domains were built around them to create a block.

3. Difficulties

a. General

One of the most problematic phases of this work was the task of grid generation. As was mentioned above, some aspects of CFD can be identified as an "art", and grid generation is at the top of this group.

The difficulties can be divided into three main categories:

- Internal inherent bugs,
- Difficulties caused by complicated geometry,
- System problems.

Concerning the first category, we have been in communication with the GRIDGEN developer, and have described the problems and irregularities that were identified during this research. Since the current version under development is No. 11 and no technical support was provided for the NPS version (No. 9.6), we had to reconcile the two versions and find ways to overcome the obstacles, as will be described below.

b. Internal Inherent Bugs

(1) Grid collapsing. Most of the grids used in this research were of the "O" type grid, which is usually used for symmetric objects. All "O" grid connectors are located on one surface. In a few circumstances, the grid/block collapsed to a zero volume configuration and the work had to be redone.

This phenomenon occurred in several cases: the first case happened while attempting to save a block after it was created by adding six domains

together. The "save" command caused the grid to collapse. The solution to this problem is to create the six domains first, save them as a GRIDGEN/ASCII file, reload this file, and create the block. No logical reason was found for either the problem or the solution.

The second case happened while attempting to modify the grid (e.g., to redimension or redistribute a connector). This caused the grid to collapse as well. No solution was found in this case, except to create the block again after modifying the connectors/domains as connectors/domains files (for that reason a copy of the connector and domain files must always be saved).

(2) Joining domains. The basic requirement for joining two grids in general, and particularly two domains together, is to maintain the grid dimension compatibility of the domain connectors being joined. However, in a few cases it was found that, although the requirement was fulfilled, the code refused to join or agreed to join the unwanted connectors (e.g., in case of multioptions of joining configurations). The solution for this is to create one of the two domains again. This time, the connectors should be added in the direction opposite to the original order, i.e., if the original domain was created by adding the connectors in clockwise direction, it will be created by adding the connectors in the counter-clockwise direction. Otherwise create one of the domains again, which, this time, should be started with another connector than was originally started.

It must be mentioned that it is written specifically in the manual, and the developer insists that the domain generation is insensitive to the order of adding connectors, but, as detailed above, the results of this research were otherwise.

(3) Creating blocks. There are cases where an attempt to create a block fails and a "wrong topology" error message is given, even though the block is made of perfectly compatible domains in dimension aspect. In this case, it is recommended to

check the direction of ξ,η,ζ in each opposite domain, even though, the domain characteristics are not supposed to be sensitive to the direction of creation. If the direction of one of ξ,η,ζ is opposite to the direction of the appropriate in the opposite domain, the code might refuse to create the block due to this reason and one of the domains in the couple has to be rebuilt in the opposite direction.

c. Difficulties Caused By The Complicated Geometry

The fewer the grid blocks involved in the analysis, the less complicated the analysis, but, on the other hand, the less accurate.

Since the major configuration investigated was a combination of a missile and a pylon, an attempt was made to create one block that would model the missile and the pylon as well. The main reason was to prevent, at least in the beginning of the research, the need for using PEGSUS which contributes a significant amount of complexity, as will be described later.

A single block grid could not be successfully generated because of the complicated configuration, mainly because of the difference in the length of the missile compared to the pylon, and to the curvature contour of the pylon, which caused a very dense grid in a certain region of the pylon [Fig. 3].

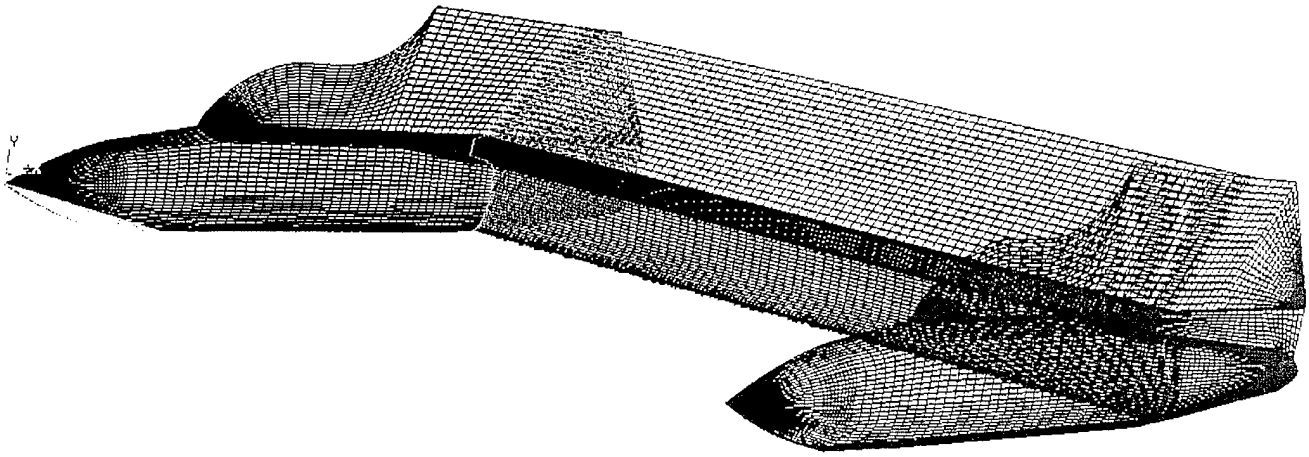


Figure 3. A dense grid structure of a joined missile/pylon block.

The phenomenon that occurred due to the very dense grid regime is the "Negative Volume" cells computed by OVERFLOW. When the flow solver identified negative cell volume, once it had read in the grid, the flow solution was not advanced and the code stopped with an error message. Using the smoothing tool of the elliptical solver and a very coarse grid finally produced a grid that was acceptable by OVERFLOW, but was physically worthless [Fig. 4].

Fig. 3 presents another attempt to get rid of the negative cells. In this case, the investigated domain was copied and the copy was placed next to the investigated domain in a way to assure that all the grid lines are parallel and, hence, no negative cells can be created. Although, theoretically, negative cells should not be in this configuration, they appeared.

d. System Problems and Difficulties

(1) Segmentation fault. The main system limitation of GRIDGEN processing is "Segmentation Fault" errors which are caused by the system and probably have no direct connection to GRIDGEN. They are caused by the network's inability to handle heavy files. These phenomena occur when trying to load a certain file (grid), or even when doing a simple action such as domain creation. The code is killed and a "segmentation error" is given. Usually it takes a while to get rid of these segmentation faults, since they are a problem with the network rather than the code.

(2) Problems caused due to file storage in the Cray. Since most of the research was done using the Cray, and due to the large grid and solution files (order of 20M for a file), the files were stored on the Cray disks and tapes. To efficient the capacity management of the files on the Cray, the files are migrated after a certain time to storage tapes (usually after 48-72 hours if the files were not used). It was found that in some cases, due to network difficulties, files that were stored on disks, could not be read

properly from the Cray by simply changing directory to the Cray from the SGI machines. Those files were loaded as corrupted files, embedded with symbols not written in the files originally. In those cases, the only solution is to transfer the files to the local machine (SGI) by file transfer procedure and then use them. Another difficulty caused by this "migration" method is that the time to load a migrated grid or solution file is very long, and can take 10-20 minutes. This phenomenon causes delays in the analysis process.

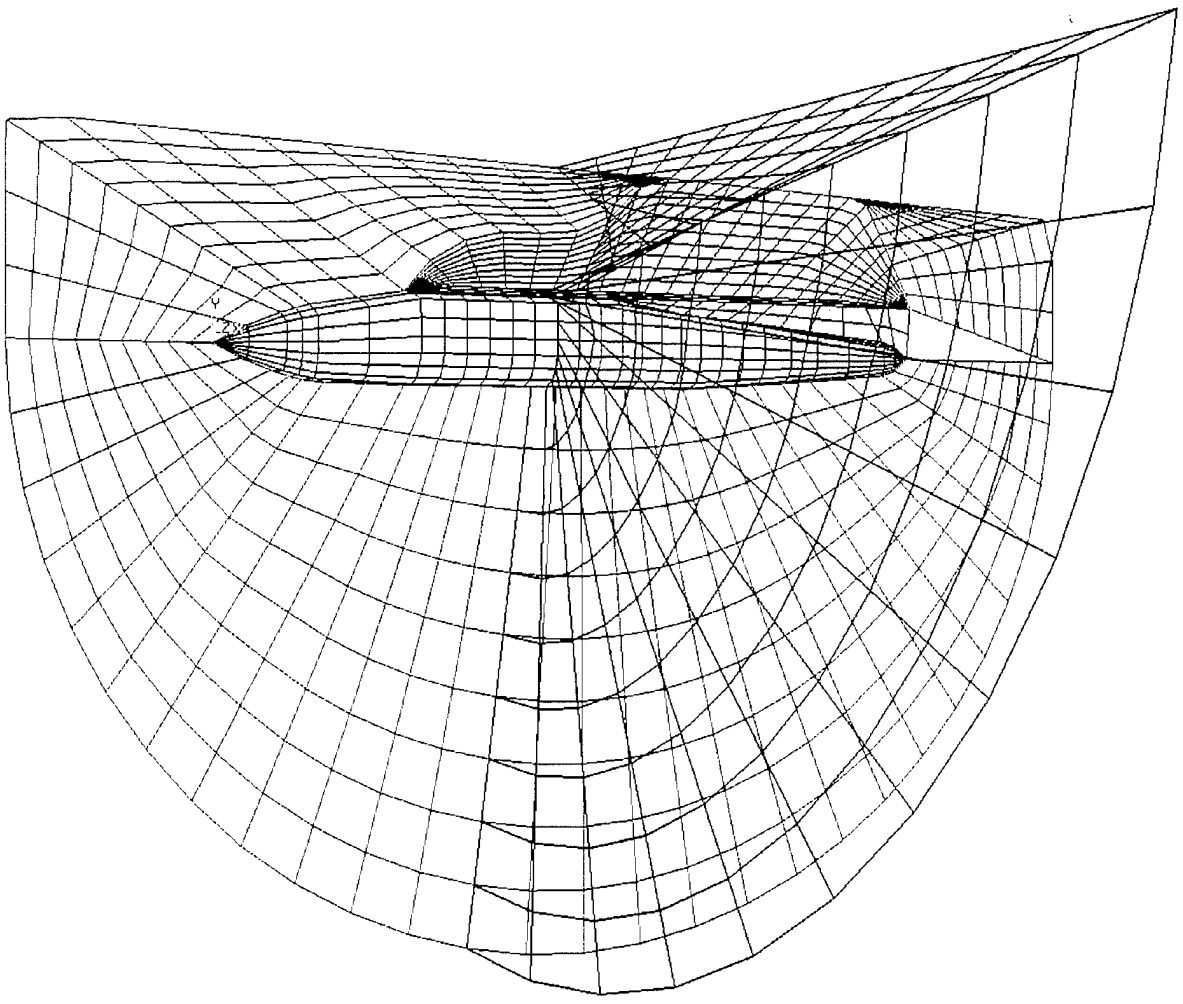


Figure 4. A coarse grid structure of a joined missile/pylon block.

C. PEGSUS

1. General

The PEGSUS code enables the investigation of complicated configurations containing as many blocks as needed, hence preventing the need to create complicated structures that can be divided into simple ones.

The current version being used at the Aeronautics computer lab is 4.0 [Ref. 9]. In this research, the PEGSUS code was used in conjunction with the flow solver, OVERFLOW, through intermediate code, MERGE41, and with the Chimera method, to interpolate between each two adjacent grids.

2. Chimera Method

The concept behind the Chimera method is given in Ref. 9 and described in Figs 5 and 6.

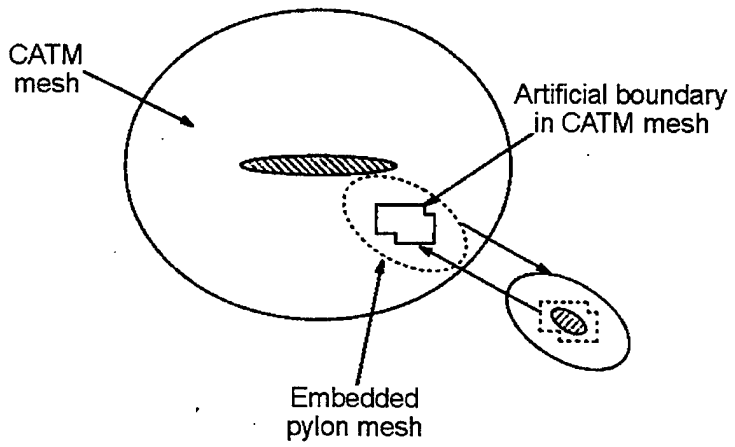


Figure 5. PEGSUS / Chimera concept - mesh to mesh communication [Ref. 9].

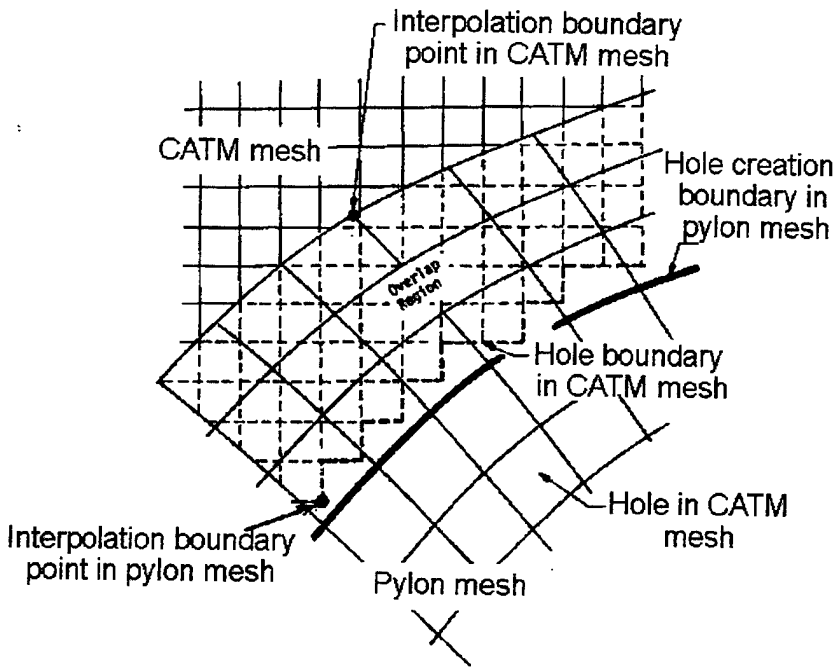


Figure 6. PEGSUS / Chimera concept - overlap region between meshes [Ref. 9].

Here are the main characteristics of the Chimera technique:

Each mesh of the two mesh couple receives and donates flow field information to and from the other. The interaction between the two meshes takes place in an "Interpolation Regime", which is embedded in both meshes.

To create this interpolation regime, two boundaries must be defined, which creates three different regimes:

- Hole point regimes where the donated grid points have to be computed, but not the accepted grid points.
- Interpolation regime where the information has been transferred from one grid to another.
- Field regime where the accepted grid points have been calculated with no connection to the donated information.

3. Technique

Once the individual grids are available from GRIDGEN, the merging of the grids can be accomplished. As a preprocedure to PEGSUS, one has to run the RMG2PEG code (see details in Section G) to create an input file to PEGSUS: "INGRID".

An input file is created [Appendix B], which in principle, contains descriptions of the grids, including which grids are linked together, which range of grid points has to be included in the updated regime (J, K, L INCLUDE variable), and a detailed definition of the HOLE boundary. What grid the HOLE is part of (ISPARTOF) and in what grid the HOLE is made in (MHOLEIN variable), as well as a precise definition of the surfaces.

4. Orphan Points

While the main obstacle in OVERFLOW/GRIDGEN is negative volumes, the major problem found in PEGSUS is Orphan Points (OPs).

The simple definition of an OPs as given in the user's manual [Ref. 9] is, "Any boundary points that remain after the mesh priority list is exhausted are termed 'orphans'". The manual also mentions that, in the flow calculation point of view, an OPs is treated as a hole point, i.e., it does not get updated by the flow solver.

If the number of OPs is limited, one can ignore them and "pay the price" of a few unupdated grid points; however, the number of OPs is usually more significant and they must be eliminated or significantly reduced (see sensitivity study in the following sections).

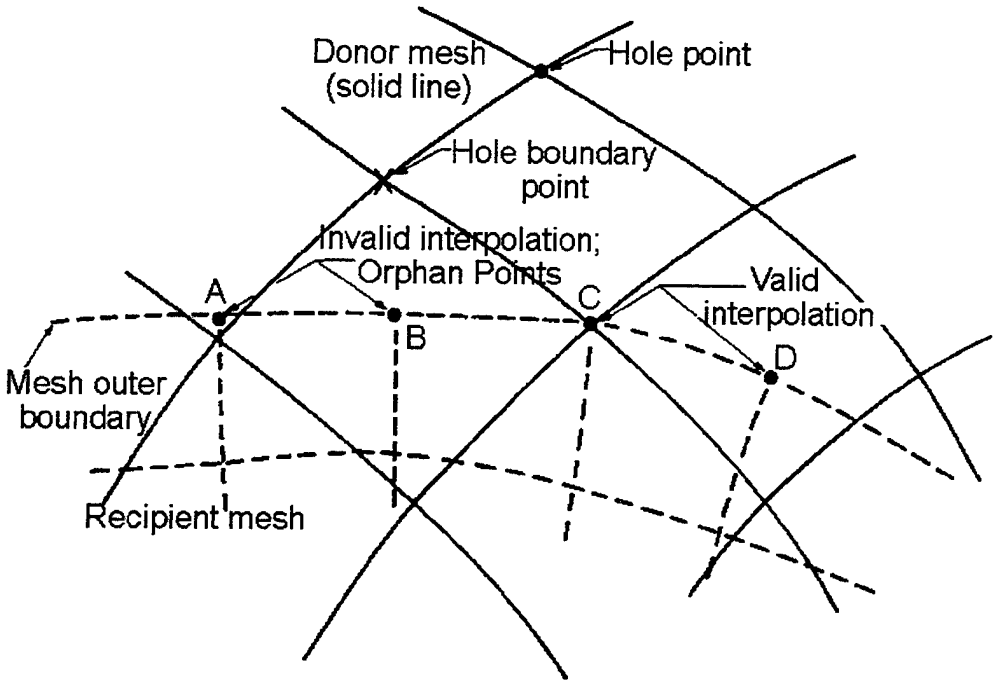


Figure 7. Orphan Points definition [Ref. 9].

To eliminate the OPs, few methods are recommended.

- Grid shifting.

As described by Barton in Ref. 7, a slight shift between the grids can significantly eliminate the number of OPs; however, this procedure might be very tedious. Since it is a trial-and-error procedure in each iteration, one of the grids has to be modified, as described above. Modification of a grid (especially using GRIDGEN) can be very time consuming.

- Enlarging the interpolation regime

In the PEGSUS input file, the inner and outer boundaries of the interpolation regime are specifically defined. In general, as the boundary increases in size, the probability of getting OPs is lower since the probability that a point won't have any adjacent point to get information from is low.

- Decreasing the range of included calculated grid points

Quite often, the OPs appear in the outer layers of the block. Since the outer layers are in the far field, one can limit the calculations to the whole grid, except for a few (approximately 2 to 3) grids on the outer boundary. It should be mentioned that those grid points are not updated by the flow solver and one has to check carefully if it does not effect the whole solution.

A sensitivity study has been done to investigate how the flow pattern can be influenced by the existence of OPs, the results of which are detailed in the multi-grid section ahead.

5. PEGSUS Output

PEGSUS creates several interpolation files, for example, "fort.2", as well as data files such as the OPs list and the IBLANK array (which contains "0" for hole points and "1" for field points).

The main file is the "grid.in" that, with the interpolation data are the input files to OVERFLOW the "grid.in" file is created actually after MERGE41 is applied on the PEGSUS output files.

D. OVERFLOW

1. General

The FORTRAN code, OVERFLOW, was developed as a flow solver code capable of using either the full Navier-Stokes or only the Euler description. Using the Chimera method in cooperation with PEGSUS and GRIDGEN, the code makes it possible to deal with complicated configurations made of several separated blocks.

At NPS, Version 1.6aw is installed on the Cray as well as on the SGI machines. The code was run in both configurations, as shown in the following sections.

2. Technique

The input file [Appendix B] includes the variables needed for the code. The detailed characteristics of the code are given in the OVERFLOW user's manual [Ref. 10]. Here are some highlights:

The principal flow data (e.g., Mach number, AOA, Reynolds number, temperature at infinity) are given in the "FLOINP" card. The next cards include information concerning the smoothing, techniques of integration, time steps, and so on.

Two important values are given in the "TIMACU" card: DT, the time step of the integration, and the CFL (Courant-Friedrichs-Lewy number).

The CFL is given by the following expression [Ref. 11]:

$$|\lambda| = \left| \frac{a \times \Delta t}{\Delta x} \right| \qquad (18)$$

where "a" is the wave speed of the following first order equation:

$$\frac{\partial u}{\partial t} + a \frac{\partial u}{\partial x} = 0 \qquad (19)$$

which is the degenerate form of the Euler basic equation without the source terms.

The convergence and stability of the solution depends, to a large extent, on the combination of these two values, DT and CFL. One of the major stages in the analysis is to find the optimized combination, which may be different from one case to another and depends on the Mach number as well.

The final section of the input file deals with the boundary conditions (BCs) to be applied. There are different BCs for viscous and inviscid solutions. The BCs can be defined specifically on certain surfaces using the initial and ending grid indices of the desired regime or on the whole grid using [-1,1] as min / max. definitions.

After the input file and the appropriate "grid.in" file (including interpolation files in the case of multi-grid) have been created, the code is run and generates several output files: "fomo.out", a history file of all the aerodynamic coefficients calculated in each iteration, "resid.out", residuals calculations (for each iteration) and solution file, "q.save" (contains information of the "Q" vector defined in equation 2 for each grid point).

In the case of a non-convergent run or a numerical problem that causes a negative calculation of the pressure or the density, a "q.bomb" file will be generated. This file contains the last valid information of the "Q" vector and can be analyzed as "q.save" being. Finally, more information concerning the values of the pressure and density is given in the "rpmin.out" file.

The "q.save" file can be visualized using FAST and the tabulated information in "fomo.out", "resid.out" files can be plotted using Matlab codes READFOMO.M and READRESID.M written by Barton [Ref. 7].

3. Useful Suggestions

a. Unexplainable Negative Cells

Cases where a negative cell error was given occurred quite often even in "restart" runs with the same input file and grid that was run the first time when the error was not given. The best solution is to copy the "grid.in" file again to the directory and run the code again.

b. Viscous Results While Setting The Viscous Terms To "F"

One must keep a consistent connection between the BCs codes and the flow definition, i.e., for example, if one sets the viscous terms to "F" (in the VISCJ, K, L parameter), and maintains the boundary condition code as a viscous one ("5", the viscous adiabatic wall), the friction data presented in the "fomo.out" file will not be zero as expected in the inviscid flow, but it will contain residual small values (2 to 3 orders of magnitude less than the pressure drag).

E. FAST

The visualization program for OVERFLOW solutions given in the "q.save" file is FAST (Flow Analysis Software Toolkit). This code, when used with powerful SGI machines, gives an impressive 3D way of visualizing flow characteristics. The code receives the grid files and the solution file as an input, and calculates the value of the desired parameter in each cell.

In case of a multi-grid run, the "grid.in" file created by the MERGE41 code must be used as the grid file (in case of Cray run, after converting the "grid.in" to a file readable by the SGI machines by using "wrgr" code) and **not** the original separated grid

files because, in the latter, the blank grids won't be seen and a non-physical solution will be presented. It should be stressed that FAST will present the solution file "q.save" on a grid file whether it contains blank points or not. Hence one must specify to read the blank points information (in the bottom of the grid file).

A few useful suggestions should be mentioned concerning the code: First, since the code runs on the SGI machines, it is very comfortable to run the initial, non-CPU consuming cases on the SGI machines and check them directly by FAST, i.e., "q.save" generated by OVERFLOW, which runs on the SGI can be read directly by FAST (as a "solution", "unformatted" file). On the other hand, in the Cray case, the "q.save" file has to be converted to a SGI readable format using a "wrq" conversion file (see next sections). Running "wrq" can take a while. Similarly, the "grid.in" generated by the SGI "xa2b" file can be read directly without any conversion, again unlike the Cray "grid.in" file that has to be converted by "wrgr" code.

Another important point is that FAST does not care about the grid distribution, i.e., one solution file can be imposed on two grids that are identical in the grid dimension but completely different in the grid distribution. This might cause confusion, unless careful follow up is being done on the files.

F. GRIDED

A very useful tool for editing grid files is the GRIDED program. This menu-oriented file enables one to change the orientation of the grid direction, flip between indices and more. This is the code used mainly to edit new blocks to a uniform configuration (in the I, J, K directions and orientation point of view) of new blocks. It's importance lies in the fact that, once the orientations and dimensions are the same, the same input files can be used for OVERFLOW.

It should be mentioned that this version of the code is probably not the latest one. In the "test" directory of OVERFLOW where test cases are given, few of the grids need to be "expanded" using GRIDED, for example, the axisymmetric test case; however, the expansion must use option "8" in GRIDED which is absent in the NPS version, and hence not all the test cases could be run.

G. RMG2PEG

This intermediate file creates the merged grid file for PEGSUS, it is a menu formatted file and uses the unformatted grid files [Appendix B].

The file simply adds together the separated grid files one on top of the other and creates one unformatted grid file called "INGRID".

H. MERGE41

MERGE41 is second intermediate file that merges the output files of PEGSUS into a PLOT3D format to be read by OVERFLOW.

The file uses the blanking scheme produced by PEGSUS and gives a "grid.in" file, which contains all the grid to be run in OVERFLOW including the blank grid points.

The hole can be seen well in Fig. 8, where the pylon makes a hole in the CATM grid, which can be identified by the hole made in the CATM grid through which the pylon can be seen.

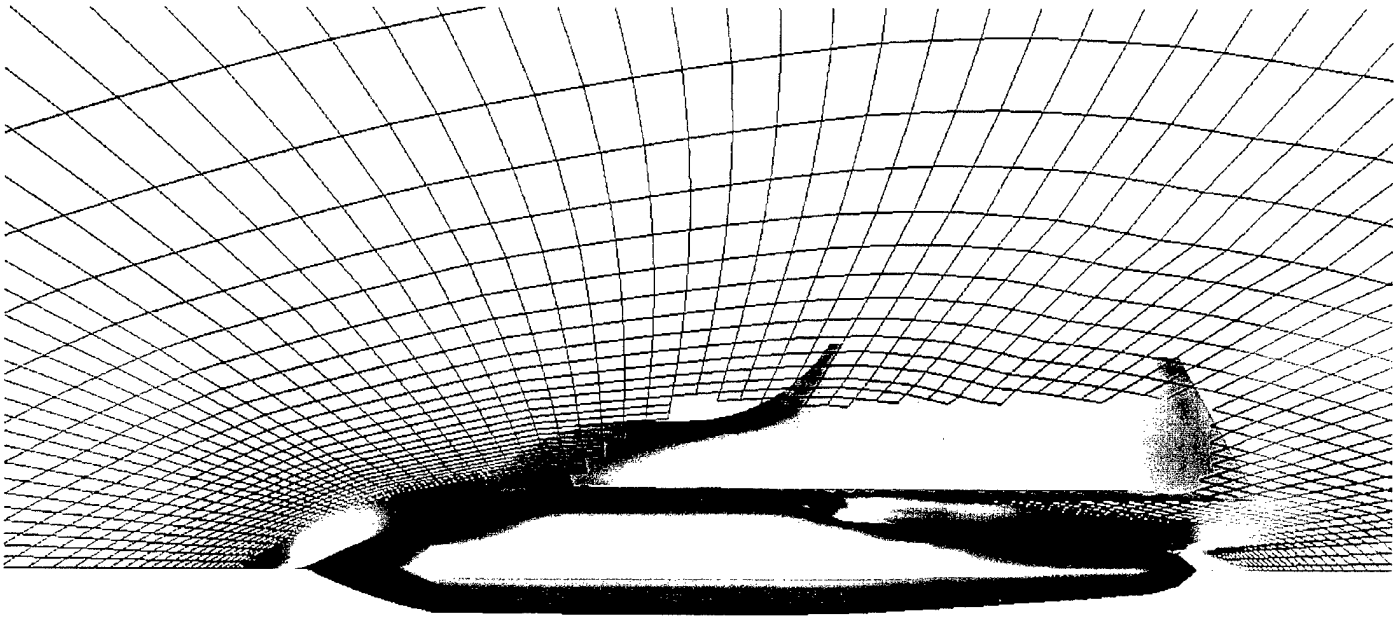


Figure 8. Hole creation in the CATM by the Pylon.

I. XB2A.F, XA2B.F - BINARY TO ASCII AND ASCII TO BINARY CONVERSION FILES

Since the flow analysis process uses formatted and unformatted files in different applications, these two conversion files change the format of the desired grid. First one has to rename the grid being investigated to "grid.for" (when running "xa2b") and get the unformatted file - "grid.in", notice that if "grid.in" file is already exist in the directory, the code will fail - make sure to erase the previous "grid.in" file. Similarly, in the "xb2a" file, one has to rename the unformatted file to "grid.in", and retrieve the "grid.fmt" formatted file.

As mentioned above, the code was installed both on the Cray and the SGI machines. In order to run the code, an unformatted grid file had to be created. The file was generated by the conversion file "xa2b.f" (Appendix B). In general, FORTRAN compilation on the Cray will create an executable file, "xa2b", which will not run on the SGI machines and vice versa. Compilation on the SGI will create an executable file for the SGI only.

J. READFOMO.M, READRESID.M

These two Matlab files were written by Barton for his thesis project [Ref. 7] and were modified in this research. The files read "fomo.out" and "resid.out" files which were renamed to "fomo_1.out", "fomo_2.out", etc. and "resid_1.out", "resid_2.out" respectively in "restart" runs. The original files could read up to four "fomo.out"/"resid.out" files. The files were then modified to read up to ten output files, and to compare between several cases (e.g., viscous and inviscid runs)

These files present a very convenient tool for graphing or processing the OVERFLOW output files; however, one can use easily GNUPLOT or XMGR, which are

also installed in the SGI machines to present the results as well. In this research, results were plotted by the Matlab files using the script file "results", which contained the procedure for creating the Cd and residuals as a function of the number of iterations and the OVERFLOW input file to have a convenient follow up of the results as a function of the input data.

K. WRQ

This file, written by Prof. Ismail H. Tuncer from NPS, converts "q.save" from its Cray format to a SGI readable file. It is a menu-oriented, robust conversion file that enables one to convert a single block as well as multiblock solution files by defining how many blocks (grids) are included in the solution file. The file creates another file, "q.cgi", which can be presented by FAST. The code, in its current format must be run on the Cray, otherwise, if has to be compiled again.

L. WRGR

Similar to the previous conversion file, "wrq", this file, written by Prof. Ismail H. Tuncer from NPS, converts the "grid.in" file in its unformatted form to a SGI readable file called "gr.cgi". This file can be loaded into FAST instead of loading all the sub grids separately. This file has it possible to visualize the grid files as they were manipulated by MERGE41, and to include the IBLANK points.

IV. TEST CASE CONFIGURATION

A. GENERAL

In order to study the OVERFLOW features and setup files, few test cases were modeled. The models description given in Appendix A, including a description of the model and the reasons for developing it.

B. THE MODELS

The first test case model (test_4_100.grd) was a symmetric model made of two rounded slices [Fig. 9]. Few cases were run using this grid, but the solutions were not satisfactory. The assumption was that the sharp corners created by the investigated domain with the symmetry upper and lower planes may have caused difficulties with the symmetry boundary conditions. The reason for it is that in case of symmetry boundary conditions, the flow information on the symmetry plane is being evaluated according to the two adjacent planes (one from each side of the symmetry plane), and once the corner is too sharp, difficulties in the interpolation between the two adjacent planes are being found in the flow solver. For that reason, an axisymmetric test case was modeled [Fig. 10], and the solutions improved significantly. The grid distribution was carefully done, since it was found that using a grid that is too coarse near the surface will cause a negative pressure drag coefficient (Cd). Hence it is important to make the grid distribution as dense as possible in the boundary layer regime. On the other hand, a too dense grid caused negative cells. For example, GRIDGEN/OVERFLOW could not handle the grid (bb_dense_4_100.grd) that its critical grid distribution on the "stings" domains was as follow - ΔS initial =0.01, ΔS final=150 even with 10 iterations of elliptical solver, the same grid with ΔS initial=0.05, ΔS final=100 and two elliptical solver iterations was successful.

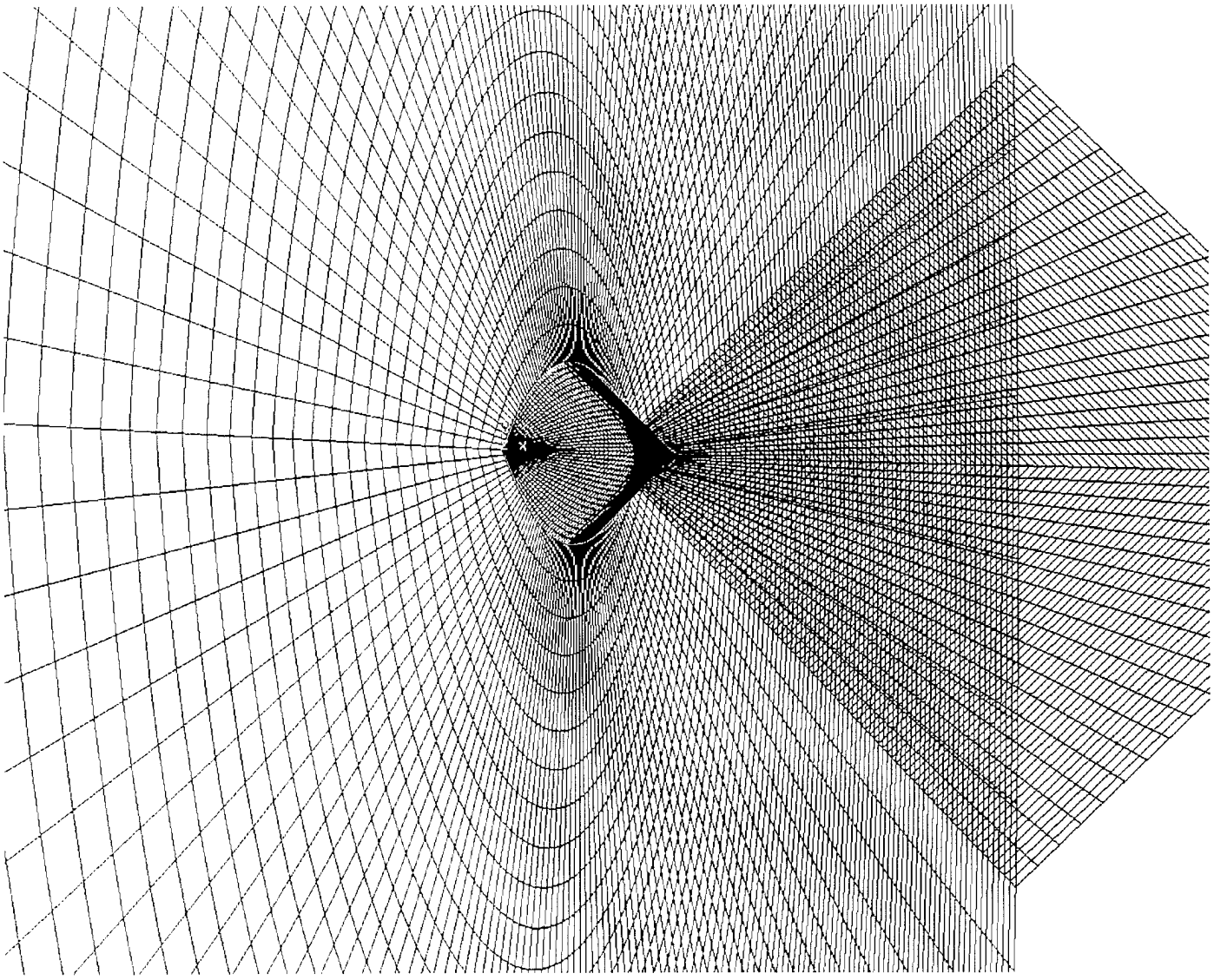


Figure 9. Test case - symmetric made from two slices.

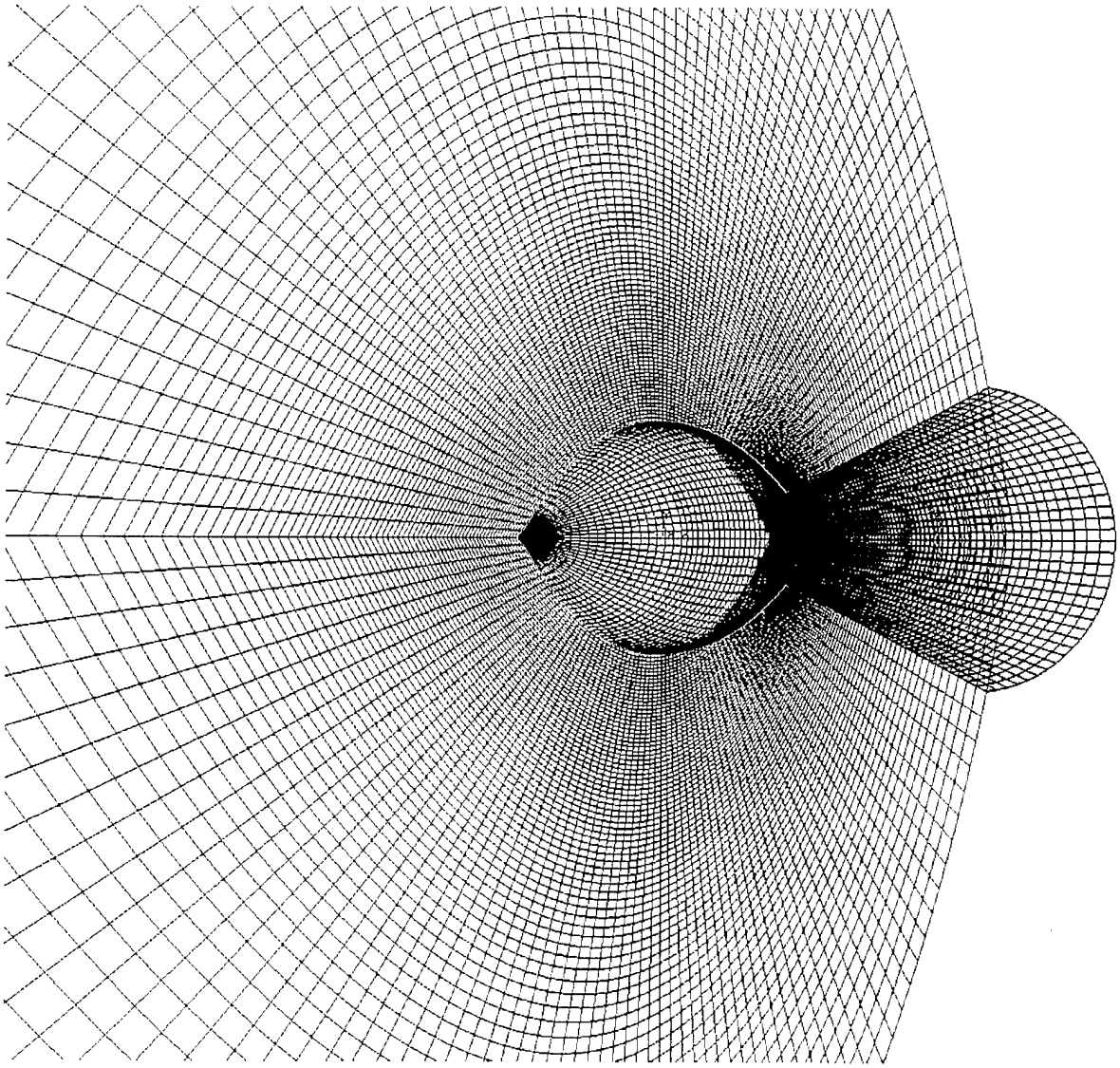


Figure 10. Modified test case - axisymmetric configuration.

C. OVERFLOW

As mentioned above, the test cases were developed to study the OVERFLOW features, and few principals ideas were found that have to be mentioned:

1. CFL, DT And Convergence

The values of DT and CFL are critical parameters to the convergence process of the code. The sensitivity to the values of the CFL and DT was carefully studied, and the OVERFLOW user's manual [Ref. 10] provides some recommendations concerning the initial values for CFL and DT, which depend on the Mach number regime. It was found that, as mentioned in the manual, a small value of DT must be used for the first trials, at least for the first few tens of iterations. Then, if the solutions seems to converge, the DT can be increased gradually.

2. GRID Topology

a. Grid Points Distribution

Besides the importance of a dense grid near the body surface, it was found that an insufficiently dense grid can cause such effects as negative Cd, and a lack of convergence, besides non-physical solutions.

b. Symmetric Blocks

When using an "O" type grid, one must not forget to build the grid in a geometry that makes it possible to use symmetry BC; i.e., in the case of symmetric BC, there must be a source of information for the surface on which the BC is implied.

Therefore, the grid must continue beyond the plane of symmetry, in other words two extra planes must be added on the other side of the symmetry plane (see Fig. 11).

D. CONVERGENCE

Convergence of the OVERFLOW solution, unfortunately, does not imply a valid solution. In many cases, such as the negative Cd cases, the solution converged rapidly, but the results were wrong.

The main parameters that control the convergence are the time step and the CFL. Running the first iterations with small values of DT (approximately 0.001 to 0.005) is recommended while leaving the CFLMIN as 0.0 (default). The OVERFLOW manual recommends [Ref. 10, p. 34] that CFLMIN=5 for subsonic until it becomes low supersonic, and that the starting value of DT=1 sec for 50 iterations, which will be decreased by a factor of 10 until the solution ceased to "explode". One must look at the behavior of the residuals in order to see if it tends to grow or not.

As the time step decreases, with no connection to the convergence, the ΔQ value decreasing. (The 8th parameter in "resid.out" represents the residuals, which are different in the "Q" vector from one parameter to the next). Hence, one can be confused and get "good" values of ΔQ (approximately 10E-07 to 10E-08) if small values of DT are implied whereas the solution really did not yet converge.

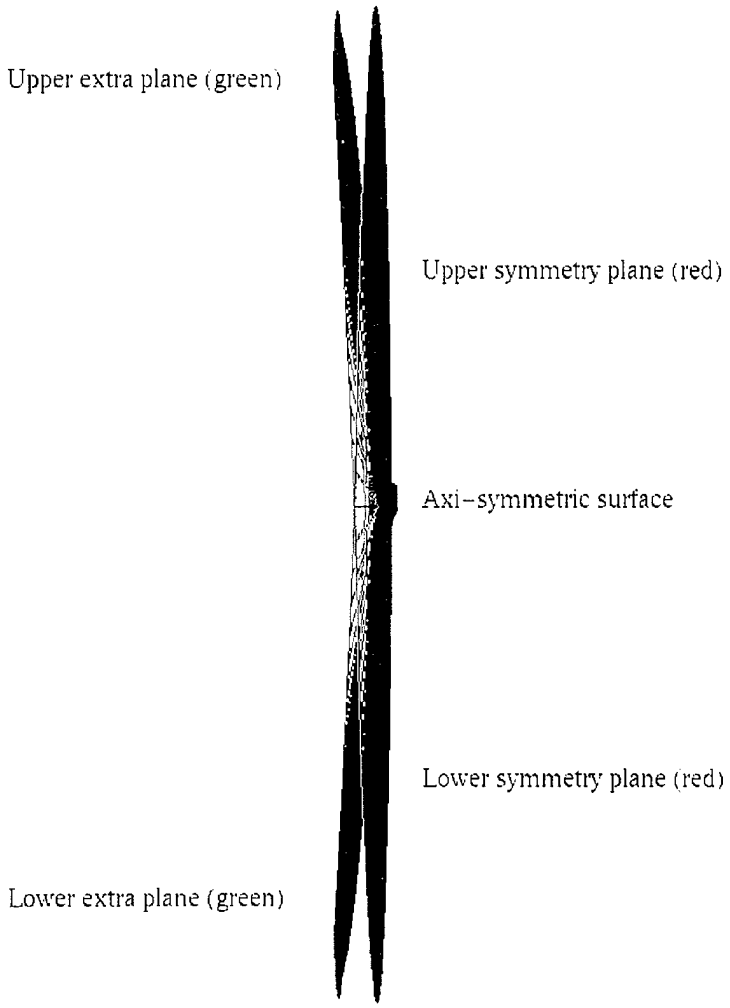


Figure 11. Two extra planes added to achieve symmetry boundary condition.

V. JSOW/CATM BODY CONFIGURATION

A. GENERAL

The basic configuration in the analysis is the missile itself, without an additional pylon or wing (Fig. 12). The analysis was begun by using the geometry provided by the contractor (TI). The geometry was given at first as a set of drawings, since the accuracy of the downloading of measurements from the draws is limited. A numerical data set was generated and a cross sectional file was sent from TI. The file is in IGES format, and attempts to convert this format to a readable format by GRIDGEN were not successful.

The grid cross-sections can be loaded as an IGES files but cannot be modified, i.e., in order to get a suitable grid one has to create a block around the basic missile configuration. This was found to be impossible even by intermediate codes such as IDEAS or AUTOCAD. In a parallel research that was done on the structural aspects of the CATM, the file could be read into IDEAS without much effort [Ref. 12].

An "O" type grid was created around the missile that was modified during this research (see Appendix A for details about the various blocks). The first configuration was similar to the operational configuration (160 in long) but without its wings and control fins.

After an optimization to reduce weight below 300 lb (Ref. 12) a shorter configuration was modeled. In essence, this configuration is the original configuration with a portion removed from the constant cross-section portion of the missile [Fig. 13].

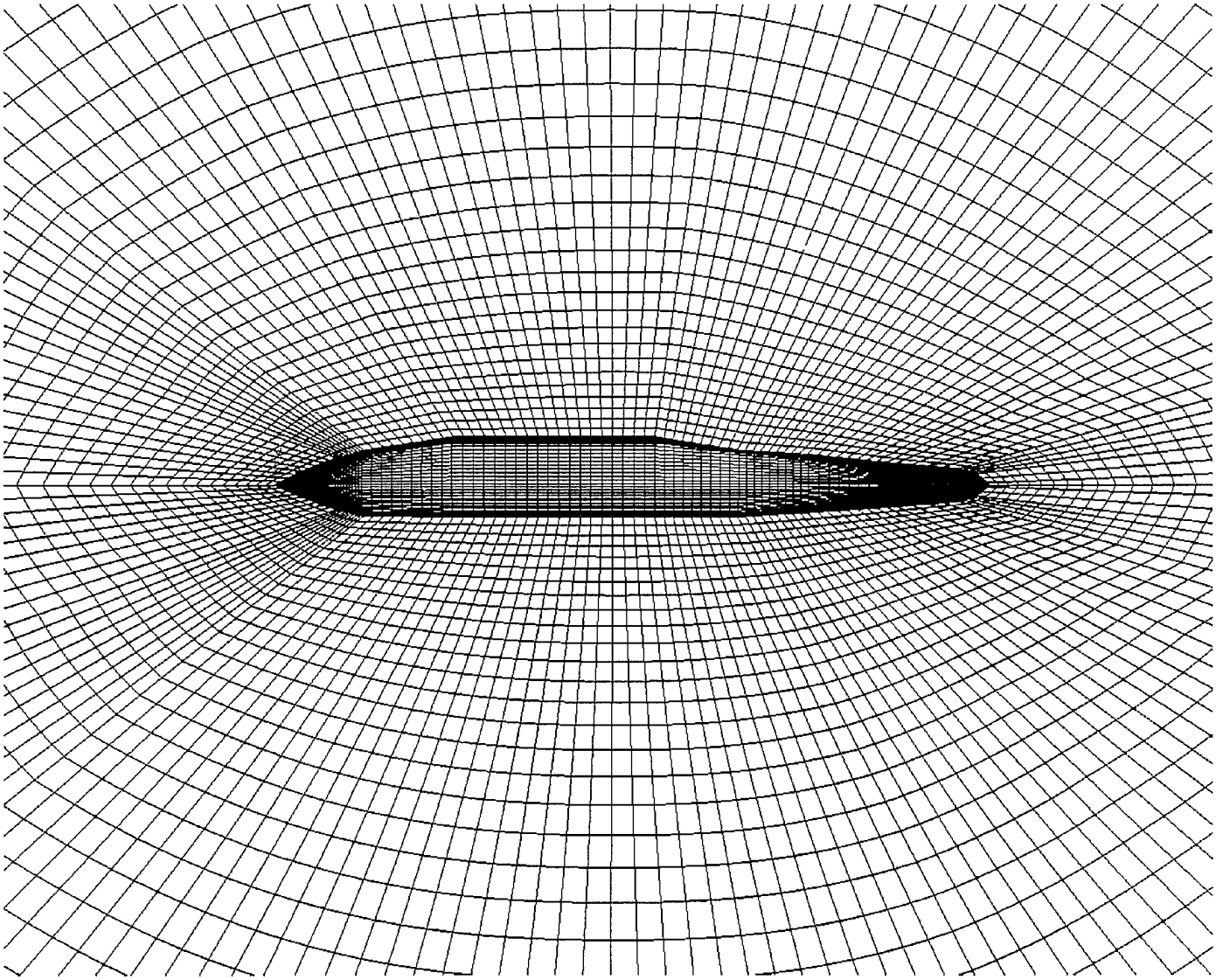


Figure 12. Original missile grid.

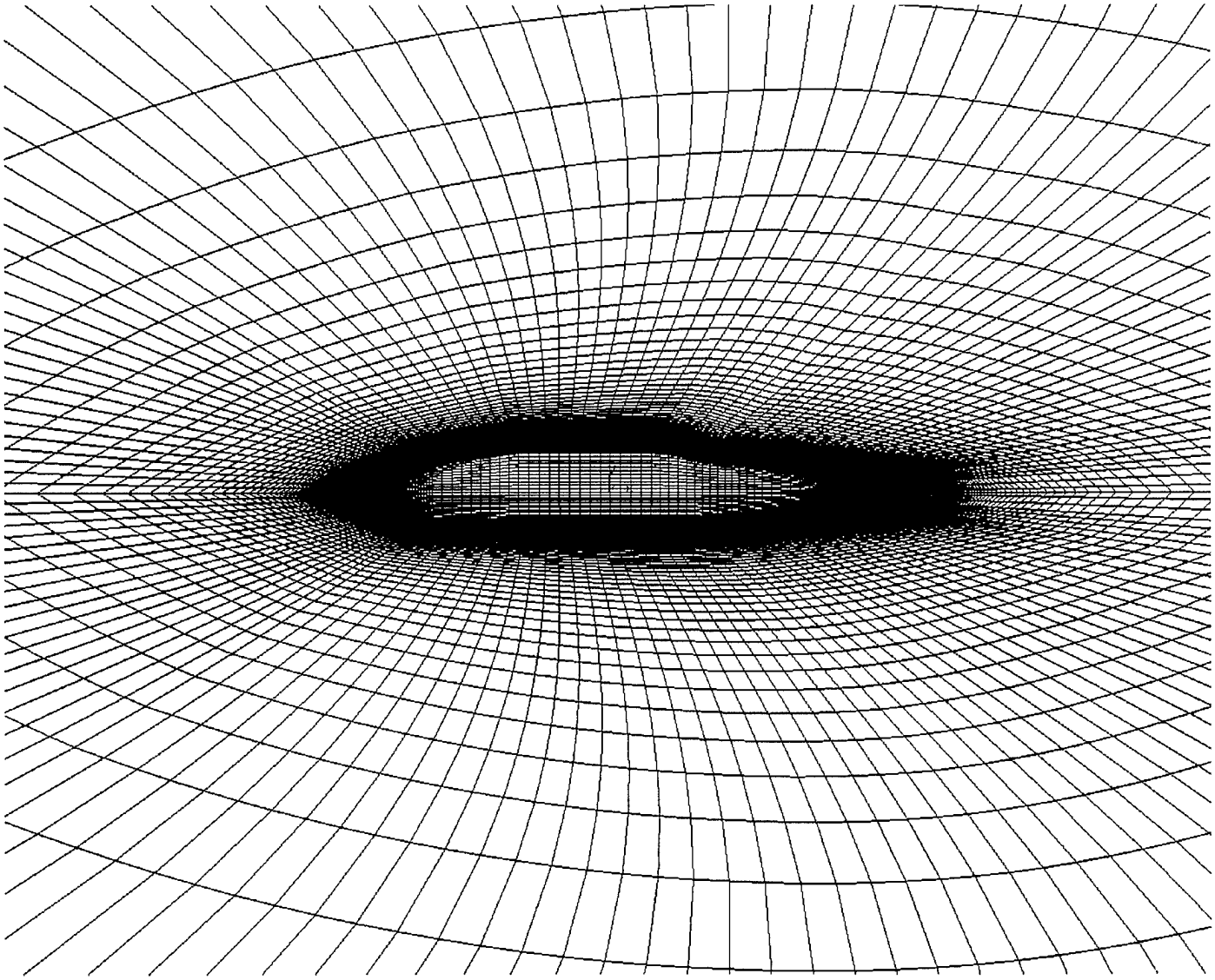


Figure 13. Modified CATM grid after was shortened to reduce drag.

B. GRID GENERATION

1. Methodology

In general, grid generation has to be made in such a way as to evaluate the flow around it, including all associated phenomena. It must also prevent the block boundaries from influencing the flow characteristics. All this, given the necessity to "save" as much as possible in the computational parts that tend to be high CPU consumers.

Assuming the flow around the body is symmetric across a central vertical plane passing across the body, through most of the research the grid architecture represented half the configuration. This assumption is not completely valid, since the actual CATM is mounted on an airplane station that has certain flow characteristics that probably can not supply a complete symmetric flow around the CATM (due to adjacent stores, aircraft structure etc.). This approach is well accepted in CFD and applied to the modeling of a full aircraft as well [Ref. 6].

The first blocks generated were relatively small compared to the dimension of the missile, i.e., the diameter of the outer domain was approximately twice the CATM length. These configurations caused wave reflections as described further on. Later, the relative block size was modified.

2. Grid Lines Distribution

In general, it is obvious that the best CFD results are achieved with a smaller individual grid cell size to achieve smoothness in the flow calculations; however, since the computations use a lot of computer resources, the dense grid orientation is important.

Dense grids have to be designed in regimes of rapid changes [Ref. 6], e.g., in the CATM case, the regime near the body boundary where the boundary layer influence is major (Fig. 14). In other cases they must be designed near sharp corners regime (not found in CATM case).

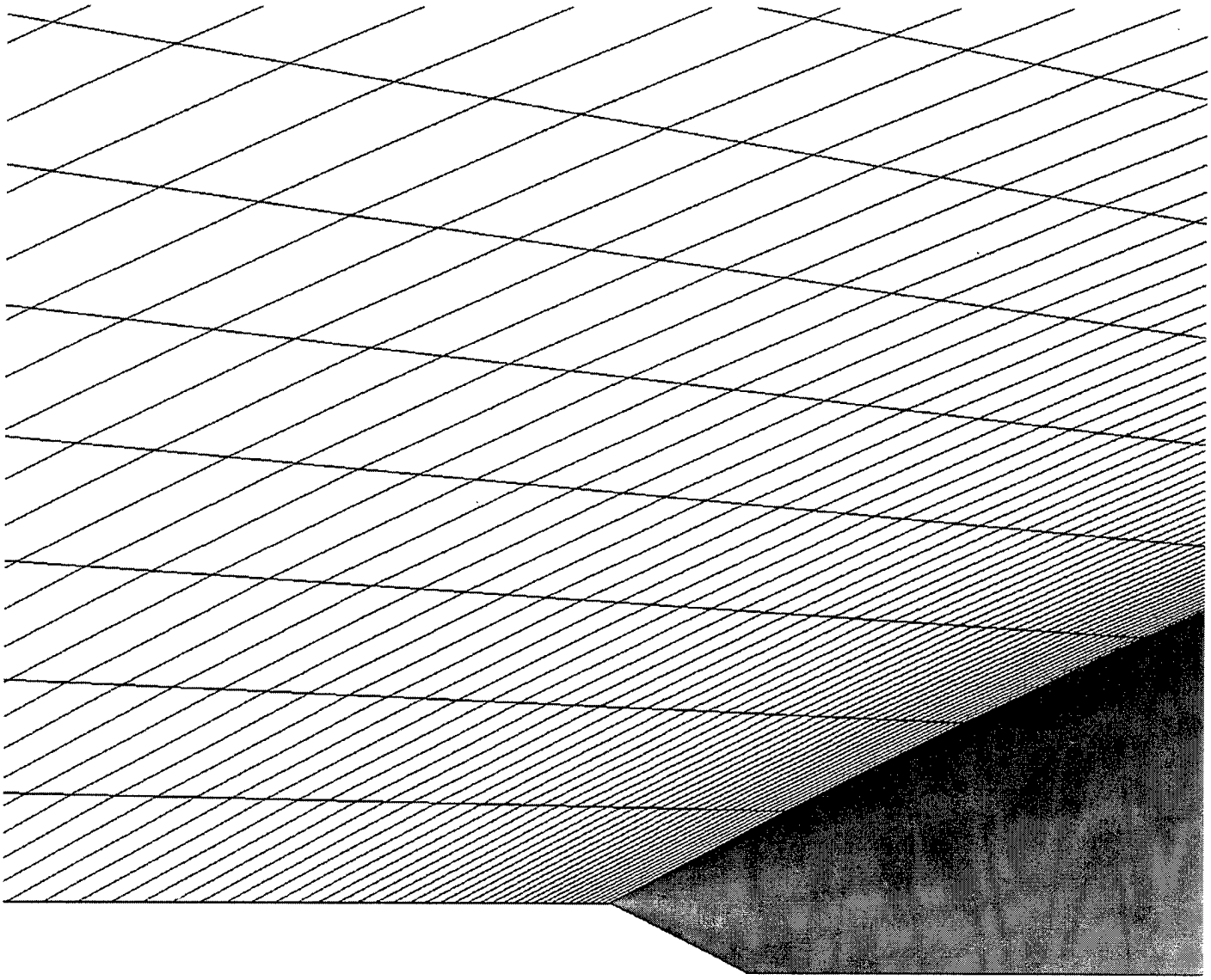


Figure 14. Grid structure of the CATM - zoom in to the boundary regime.

Another situation where a dense grid has to be applied is to create uniformity in the block. In order to prevent phenomena such as grid surfaces bending, the grids can be redistributed on the main domains. The advantage of this, besides smoothing the grid shape, is preventing negative cells in the grid that are caused by extremely bent gridlines.

Technically, the grid distribution is controlled by the "Redistribution" function in the "Connectors" menu. In principle, the code makes it possible to spread the grid points along the connector by several schemes, in our case we used the geometrical distribution which distributes the grid points by definition of the initial distance (ΔS) and the final distance between them, the gaps lengths are distributed geometrically. It should be mentioned that, if the initial definition of ΔS is "0" - the Jacobean will be zero since the volume of the first grids cells is zero. However, GRIDGEN prevents this and sets a certain small number (but usually not small enough) to get a valid solution. Hence, one must specifically define the initial value of ΔS and make sure it is not zero.

C. FLOW ANALYSIS

1. General

Getting a solution in a single grid case is relatively simple, but the validity of the solution has to be carefully assessed. The CATM model was run in a Mach range of 0.5 - 1.8. As the Mach number became higher, the solution converged more rapidly due to the fact that the influence regime of the flow field became more limited. The Mach angles in supersonic flows are small and the interactions with the block boundaries are almost invisible (Fig. 15, 16 and 17). In the case of the lower Mach numbers, the convergence is slower but can still achieved.

In general, the CATM body investigation was the base study for more complicated configurations. Therefore, a wide variety of flow conditions were applied such as changing the Mach number, angle of attack, and solution type (Euler, vs. NS).

2. Boundary Conditions

Overflow offers a variety of BCs for Euler inviscid analysis as well as Navier-Stokes solutions. A detailed table containing all the available BCs is given in the OVERFLOW user's manual [Ref. 10], Table 3.3.

The BCs that were used in this research are as follows (see Fig. 18):

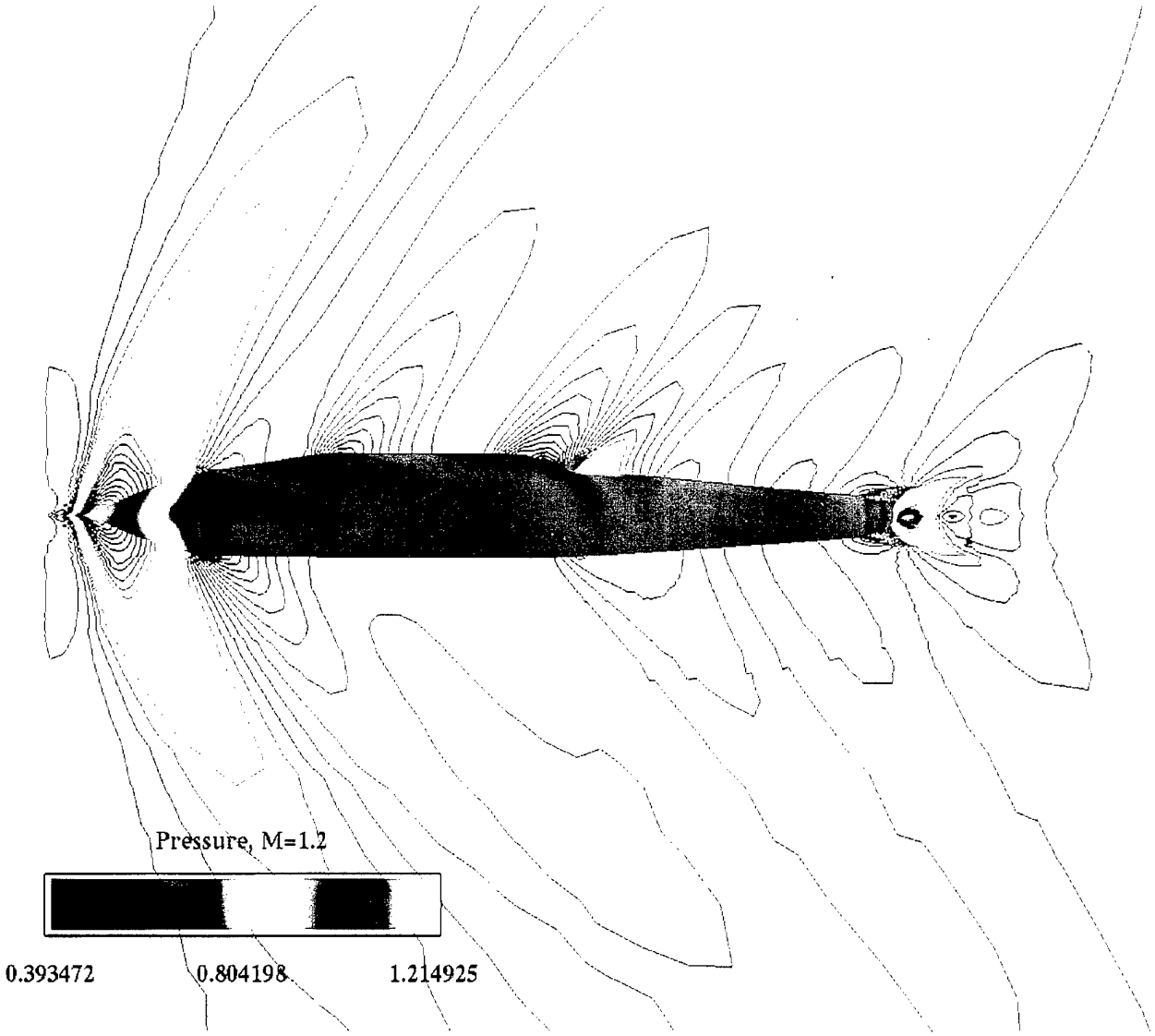


Figure 15. CATM flow field, free flight Mach 1.2.

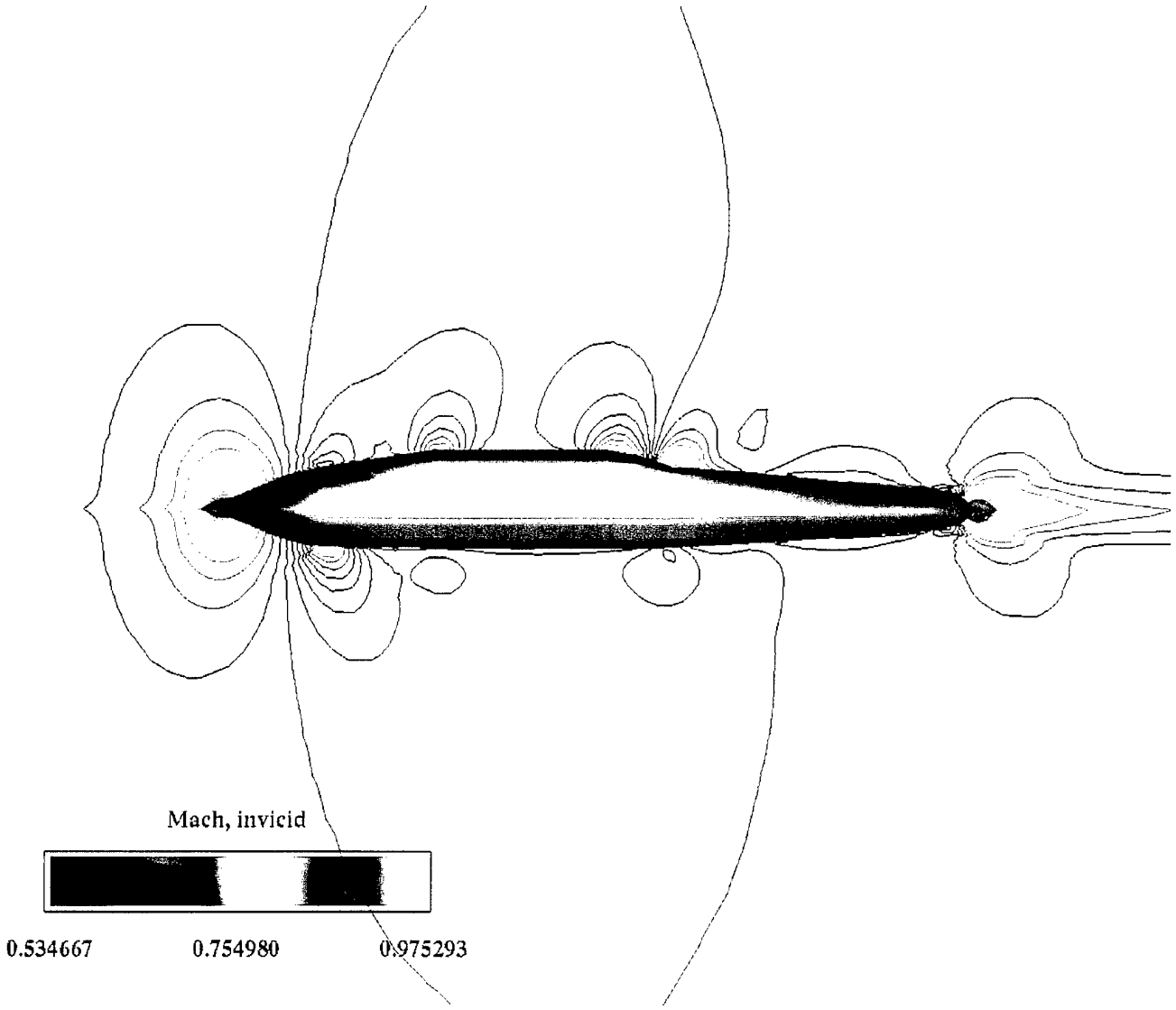


Figure 16. CATM flow field, free flight Mach 0.85, AOA=0 deg, inviscid case

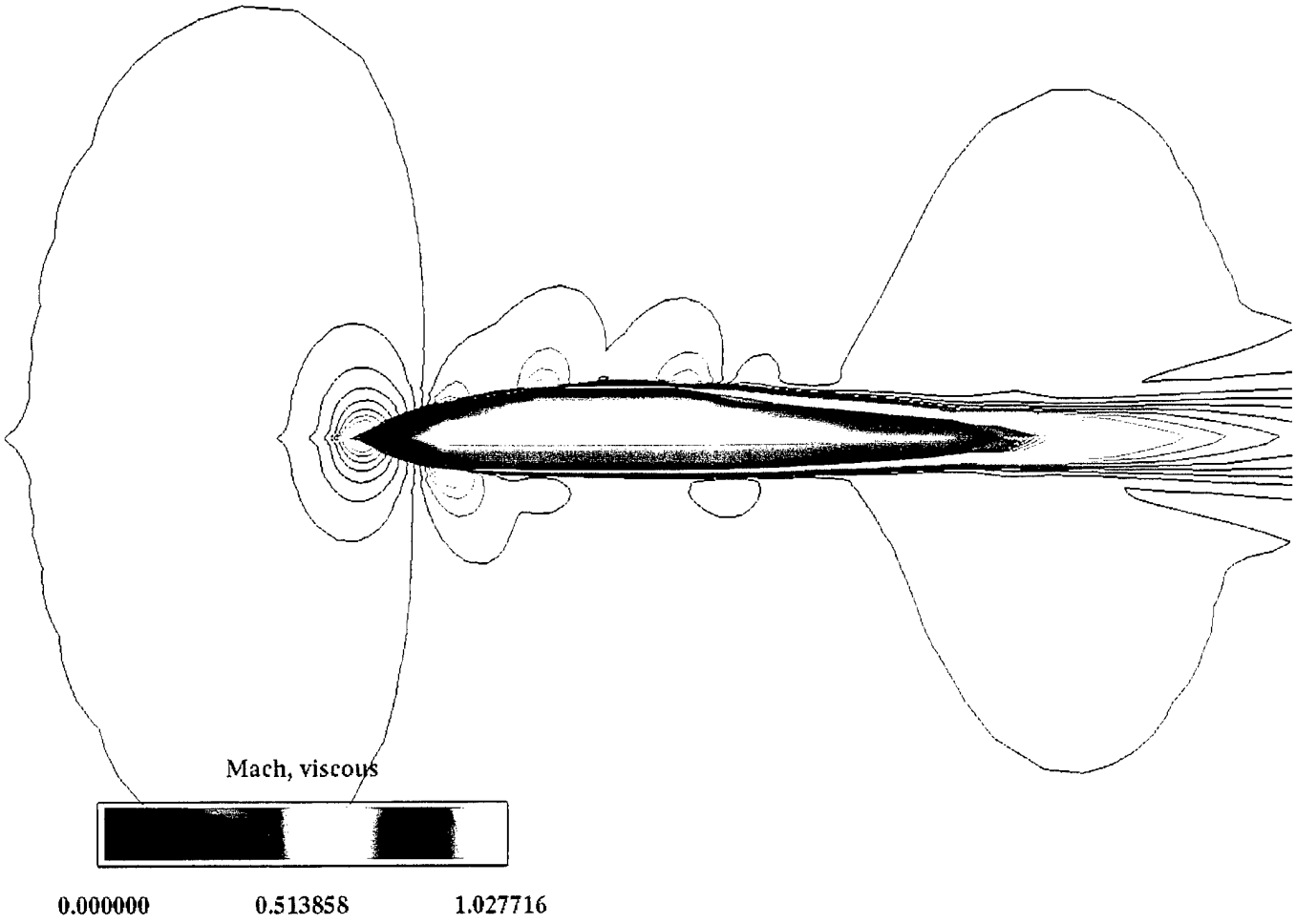


Figure 17. CATM flow field, free flight Mach 0.85, AOA=0 deg, viscous case

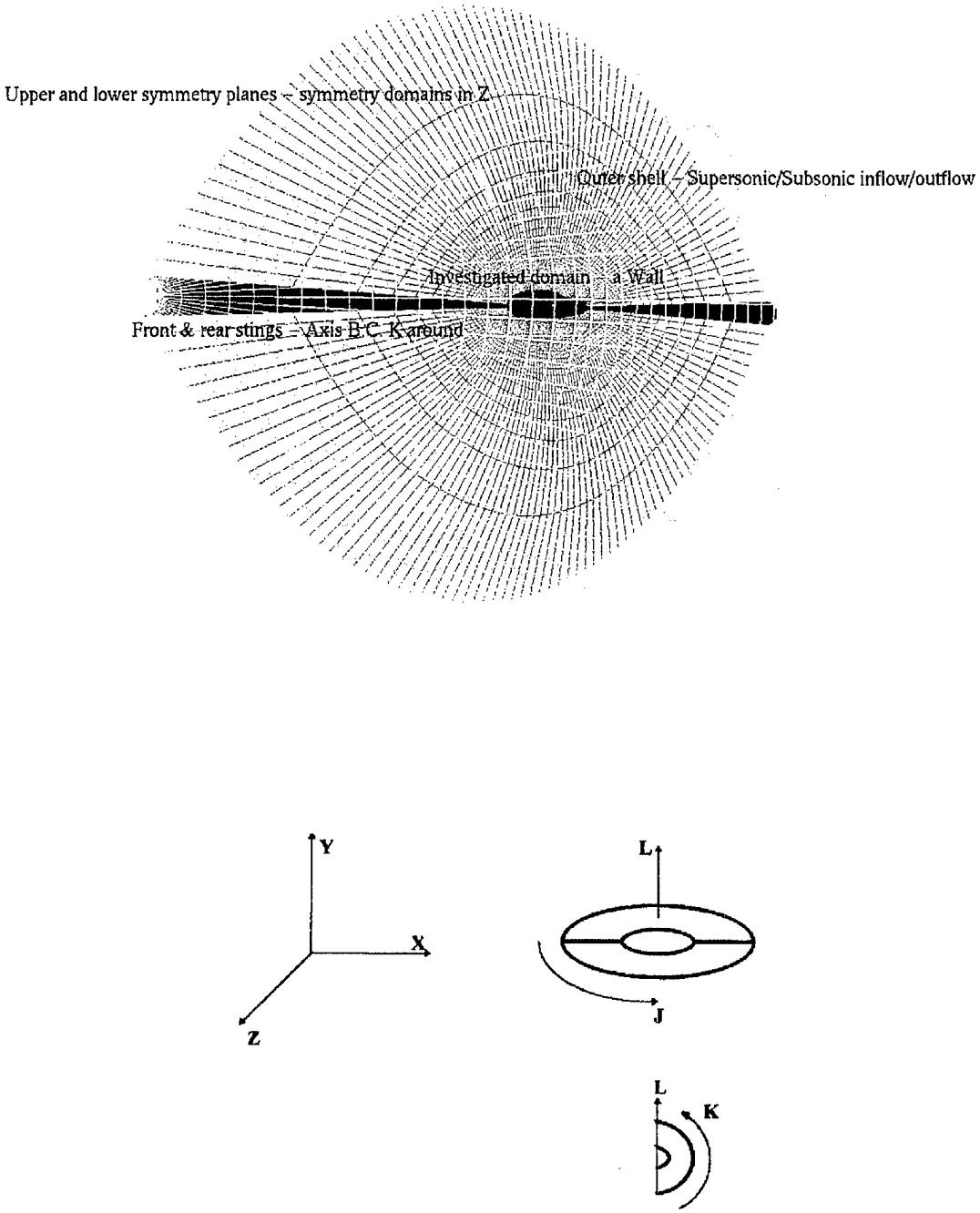


Figure 18. Boundary Conditions definition.

a. The investigated domain (domain No. 1) was defined as a wall
- Inviscid Adiabatic Wall (#1 in the table) for Euler solution.
- Viscous Adiabatic Wall (#5 in the table).

b. The two stings (Domains No. 2, 3) were defined as an Axis BCs (#15 - K around).

c. The two flat domains above (Domain No. 4) and below (Domain No. 5) the body were defined as symmetry domains in the "z" direction, #13, (i.e., the solution on +/- Z is identical). --

d. The outer shell of the block (Domain No. 6) was defined as Supersonic/ Subsonic inflow/outflow - #32.

3. Reynolds (Re) Number Calculation

In the case of viscous solutions, the Re Number must be calculated as follows:

Assumptions:

- Constant kinematics viscosity and equal to the value at $15^0 C$ $\quad \nu = 0.15 \ [cm^2 / \sec]$
- Constant temp. $\quad T = 15^0 \ [C] = 288^0 \ [Kelvin]$
- Reference length - Body diameter $\quad L = 45.2 \ [cm]$
- Gas constant for air - $\quad R = 287 \ \left[N \times M / Kg^0 K\right]$
- Heat capacity for air - $\quad \gamma = 1.4$
- Nominal Mach Number - $\quad M_\infty = 0.85$

$$\mathrm{Re} = \frac{UL}{\nu} = \frac{M_\infty Lc}{\nu} = \frac{M_\infty L\sqrt{\gamma RT}}{\nu} = 8.713*10^6$$

4. Results

a. Mach Number Sensitivity:

The nominal Mach number investigated was $M_\infty = 0.85$, which is approximately the critical Mach number. Since most of the research was done on transonic phenomena, this value was of interest.

OVERFLOW is quite sensitive numerically at $M_\infty = 1.0$. Previous researches by Biblarz and Priyono [Ref. 13] showed the difficulty of achieving good convergence at $M_\infty = 1.0$. Hence, slightly different values were given; 0.98 or 1.02.

The main reason for this sensitivity can be explained by the boundary condition implementation of the code. The code uses a method of characteristics to apply the boundary conditions. In general, as was described in Chapter II, five equations must be solved to get the 3 velocity components, the density and the internal energy. The eigenvalues of the equations are in the order of [Ref. 11]:

$$\Lambda = \begin{bmatrix} U & 0 & 0 & 0 & 0 \\ 0 & U & 0 & 0 & 0 \\ 0 & 0 & U & 0 & 0 \\ 0 & 0 & 0 & U+c & 0 \\ 0 & 0 & 0 & 0 & U-c \end{bmatrix}$$

where U is the upstream velocity and c is the speed of sound under the specific conditions. U represents one of the boundary conditions defined to the block, the inflow condition. It can be seen that there is a singularity in the case where U = c and one of the eigenvalues becomes zero.

In this research, two sets of runs were done, for viscous and inviscid cases, both of which converged without any problems. Again, the reason for the ambiguity between this research results and Biblarz and Priyono's research results might be due to grid sensitivities.

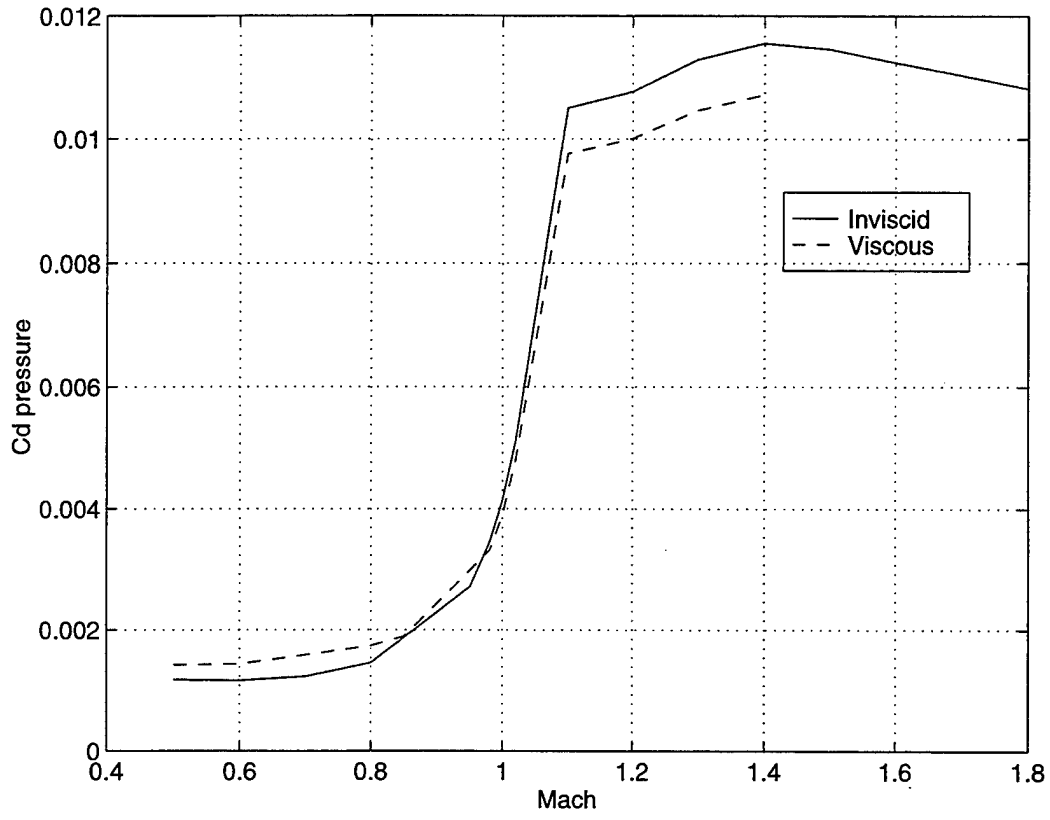


Figure 19. CATM Cd pressure as a function of the Mach number.

As can be seen from Fig. 19, the expected configuration of Cd as a function of the Mach number, where a maximum values are achieved slightly after M=1 was not achieved.

This may be due to the following:

- As was mentioned, the OVERFLOW code is sensitive to the transonic regime; hence, the results in this portion could be inaccurate.
- Although the solution behaviors of convergence and stability were perfect, phenomena related to the grid generation may have caused the solution to be distorted.

In the viscous case, no solution was achieved above Mach=1.5. OVERFLOW identified a negative density or pressure and stopped its iterations, and the "q.bomb" file was created for the last iteration. However, it can be seen that the pattern of the viscous case follows the pattern of the inviscid case.

The solution problem probably originated in the grid configuration of the CATM. Since Mach=1.5 is far above the interested Mach regime, this phenomenon was further investigated.

b. Solution Type - Euler vs. NS Solution

Both the NS and Euler solutions converged smoothly through all the investigated Mach number regimes (0.5 - 1.8). The higher Mach numbers help the convergence to be more rapid.

As mentioned, the difference between NS solutions and Euler was not dramatic; hence, most of this research was done with the Euler solutions, since they are much faster and consume less CPU. Figs. 20, 21 and 22 demonstrates the solutions at the nominal Mach number.

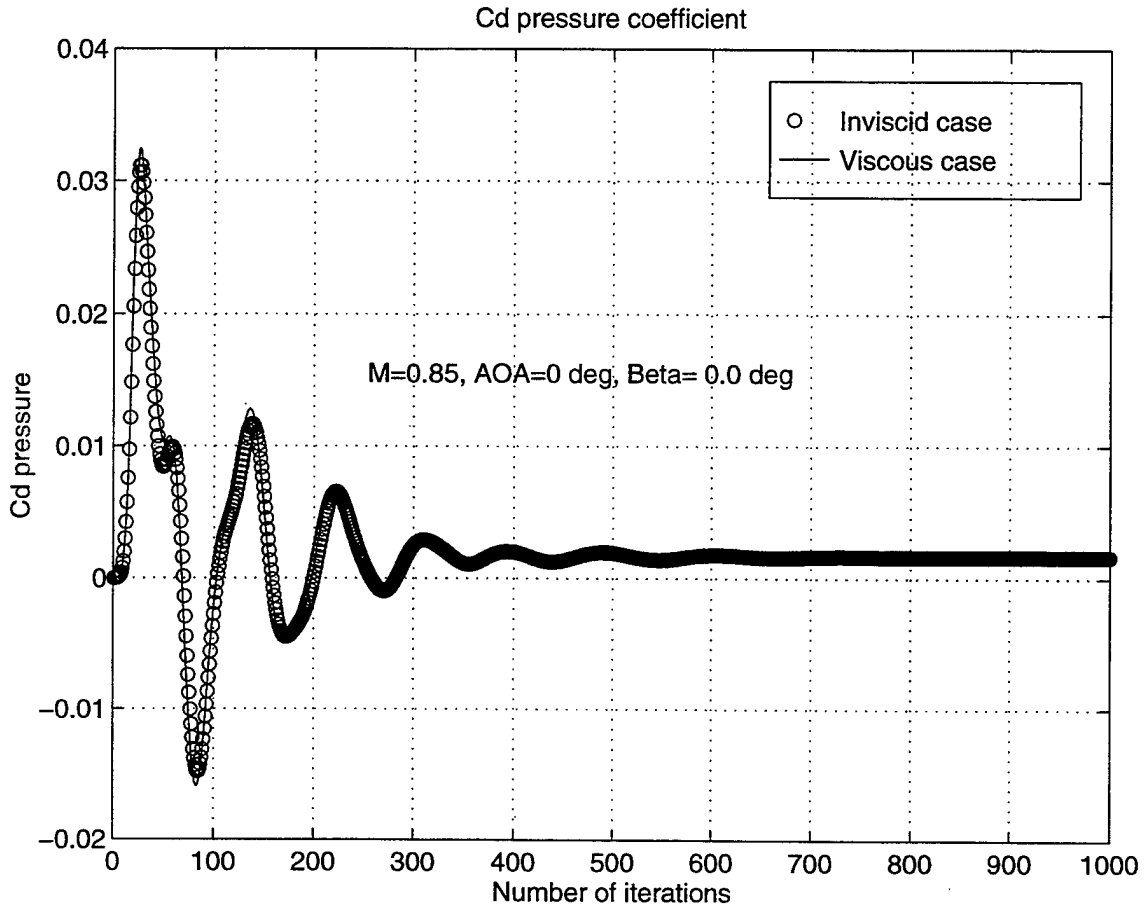


Figure 20. CATM alone- Cd pressure of Euler and NS solutions in M=0.85.

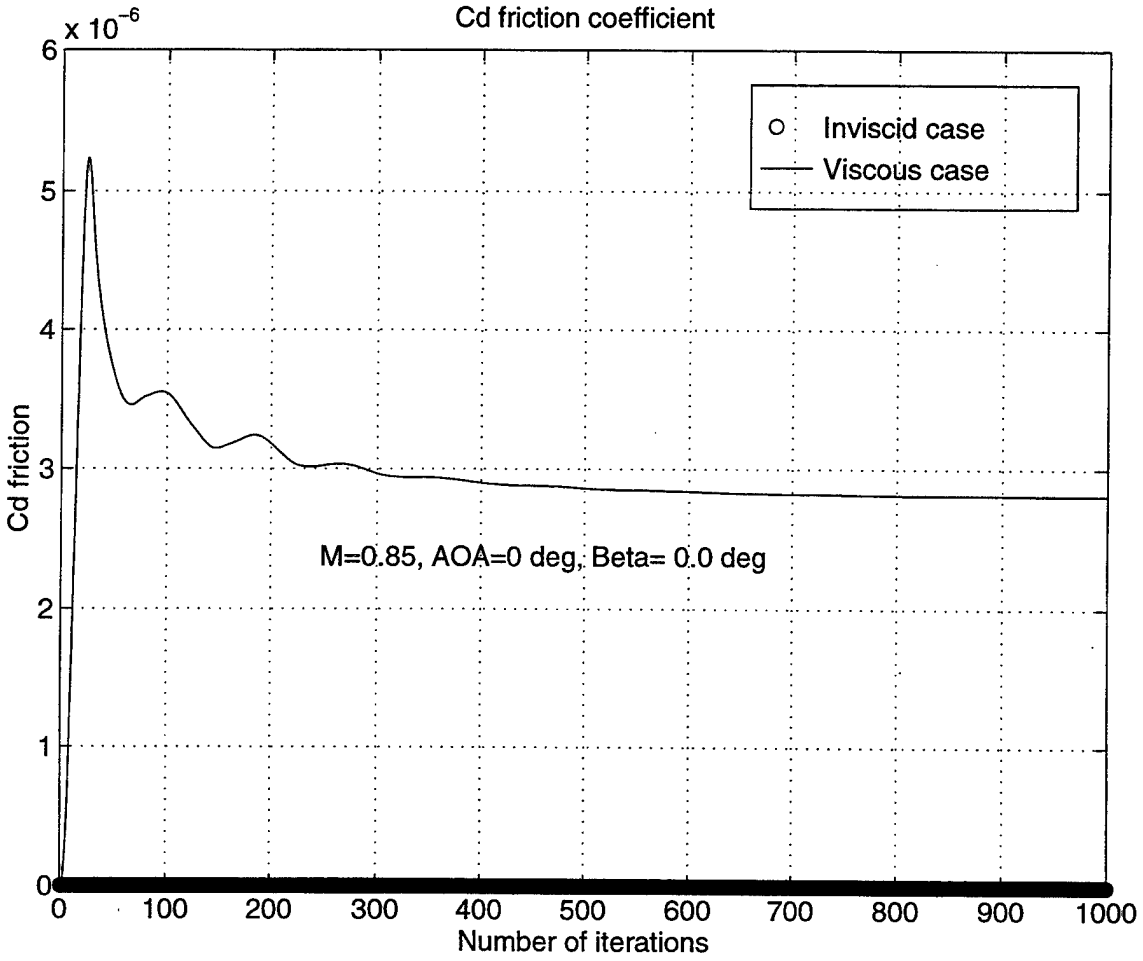


Figure 21. CATM alone, Cd friction of Euler and NS solutions in M=0.85.

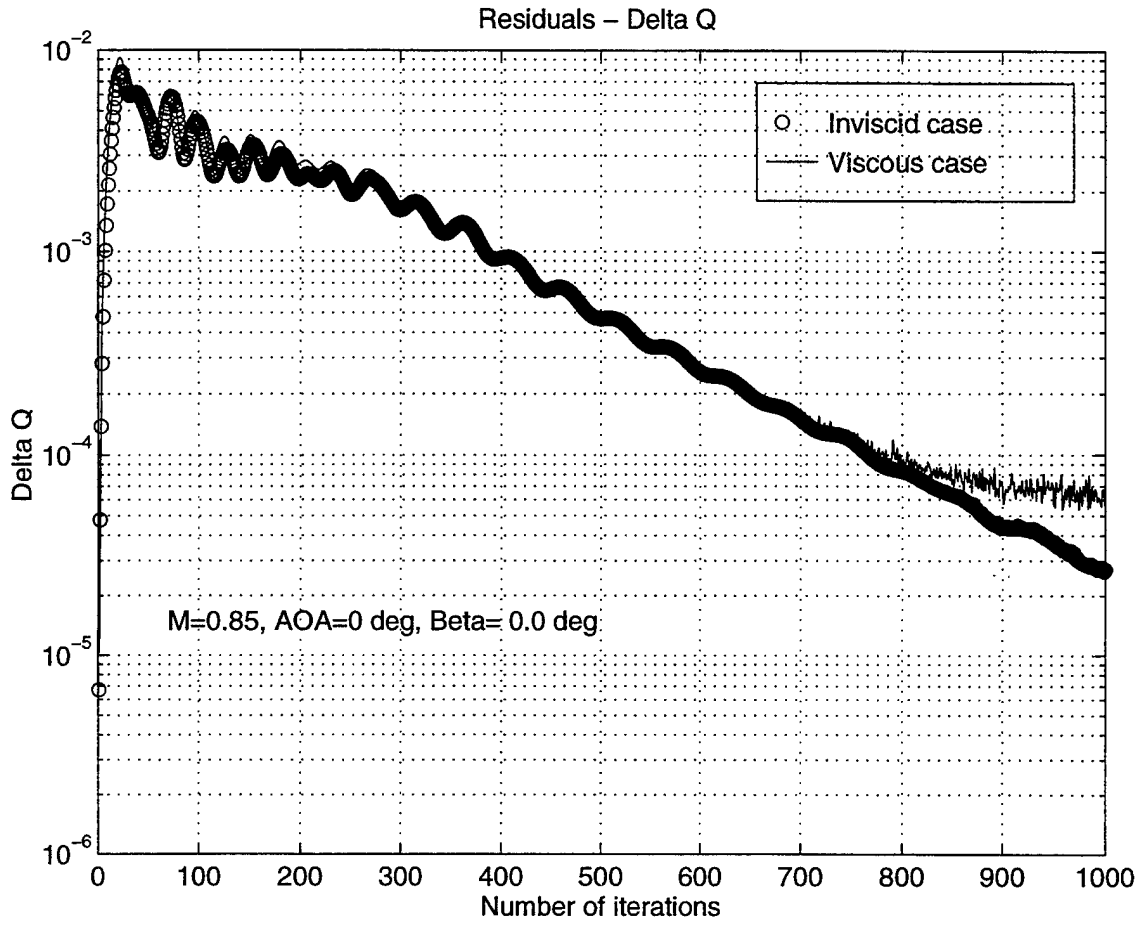


Figure 22. CATM alone- residuals of Euler and NS solutions in M=0.85.

c. Angle Of Attack Sensitivity

As a technical remark, it should be mentioned that, since the orientation of the grid as developed using GRIDGEN was in a way that the symmetry plane was across the Z axis, the angle of attack and sideslip parameters had to be exchanged. The reason for it is that the angle of attack, α and sideslip angle, β are defined in OVERFLOW as:

$$\alpha = \tan^{-1}\left(\frac{w_\infty}{u_\infty}\right) \qquad (20) \quad , \quad \beta = \sin^{-1}\left(\frac{-v_\infty}{U_\infty}\right) \qquad (21)$$

where $u_\infty, v_\infty, w_\infty$ are the free stream components and U_∞ is the total free steam magnitude, equals to $|u_\infty, v_\infty, w_\infty|$ in the body frame [Ref. 10].

Hence, the definition of the AOA in the input file of OVERFLOW has to be under the sideslip angle (BETA) and vice versa, under angle of attack (ALFA), the value of the sliding angle has to be defined.

Two main blocks were built; "jb_block11.grd" and "jb_block13.grd". The first was used mainly in the analysis, and the latter was a rotation of the first one by 90 degrees to get a true orientation for α, β (see Appendix A for details).

The sensitivity to angle of attack was investigated mainly on the CATM alone since the purpose was to analyze the influence of AOA separate from other parameters such as interference between the CATM and the missile. The sensitivity was checked in the NS viscous solution between 0 and 30 degrees. As expected, the drag increased rapidly with the AOA, see Fig. 23.

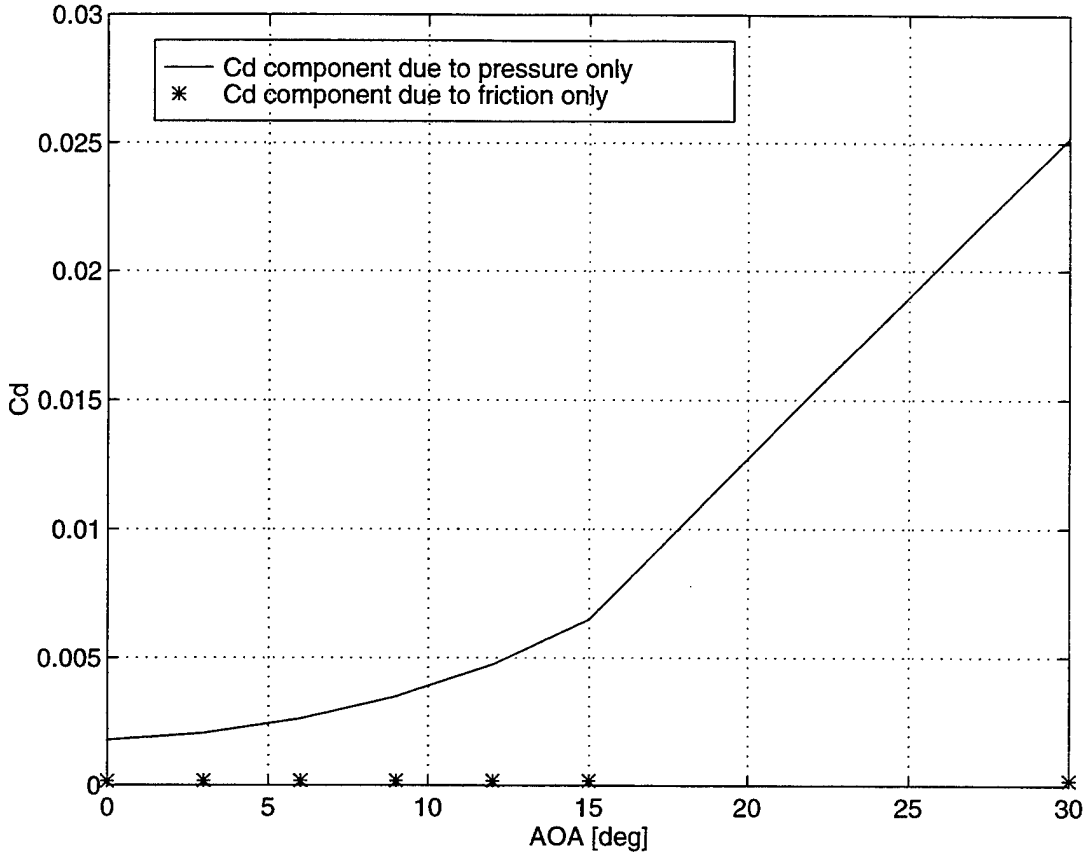


Figure 23. Cd of the CATM as a function of angle of attack.

It should be mentioned that, in cases of convergence difficulties, increasing the angle of attack by several degrease (approximately 3-5 deg) helps to accelerate the solution convergence. This effect was published in Ref. 5 as well. The reason for it is because at zero angle of attack, the vertical velocity component is zero and the flow solver finds it more difficult to calculate the flow and the solution converges slowly. In higher angles of attack, other phenomena occurs such as flow separation and the convergence might slow again. This was not investigated in this research.

VI. MULTI-GRID CONFIGURATION

A. GENERAL

To investigate the combination of the CATM and the pylon, a multi-grid scheme was developed and run in various flow conditions and at relative locations between the two grids (see the Area-Rule in the following section). In addition, a wing grid was added to investigate the influence of wing existence above the CATM/Pylon system.

It is common to use a peripheral grid (outer box) around the investigated grids in order to simulate the far field as closely as possible. This idea was applied in this research too; however, it was found that the CATM grid that was developed was broad enough so that the flow could fully conform to the free stream condition within the CATM grid itself, and an outer box was not needed. Comparison of the results with and without the outer box proved this; hence, the outer box was not used in further investigations.

B. GRID GENERATION

As a first attempt to analyze the flow around the CATM and the pylon, one grid was developed, consisting of the two configurations together. The intent was to prevent, at least for the first stages of the research, the necessity to deal with multiblock analysis which, as was described, is quite complicated.

Mainly because of the differences between the length of the pylon and the CATM and the curvature lines of the pylon, no one satisfactory grid was found. The obstacles were "negatives cells", recognized by OVERFLOW as preventing calculation of the flow by not updating these grid points in each iteration. An example of the combined block has been given in Fig. 3. In this block, as mentioned, the idea was to get rid of the negative cells by creating parallel grid lines with two identical domains: the CATM/Pylon

combination, placed apart from each other. Even this method failed, although logically it cannot create negative cells.

The process of multi-grid generation was described previously. In this research, four separate grids were developed [Fig. 24].

- CATM grid - size 80 x 39 x 75 grid points.
- Pylon grid - size 28 x 60 x 80 grid points.
- Wing grid - size 80 x 30 x 39 grid points.
- Outbox grid - size 100 x 80 x 60 grid points (not shown in Fig. 24).

The outbox grid was not used exclusively in the process because, as was mentioned, the size of the CATM grid was broad enough that the flow got to its free-stream condition far from the boundaries of the block.

One of the main issues of this research was the Area-Rule investigation. This subject will be elaborated upon in the next sections. In principal, the grid of the CATM was duplicated and shifted to get various cross-section area-distribution configurations.

C. RESULTS

The multi-grid cases were analyzed both in the CATM-Pylon-Wing configuration and the CATM-Pylon configuration only. As in the single grid, the research was done mainly in the transonic regime; hence, M=0.85 was chosen to be the main Mach number. The analysis was done both in Euler and NS solutions. Typical results are presented in Figs. 25 through 28:

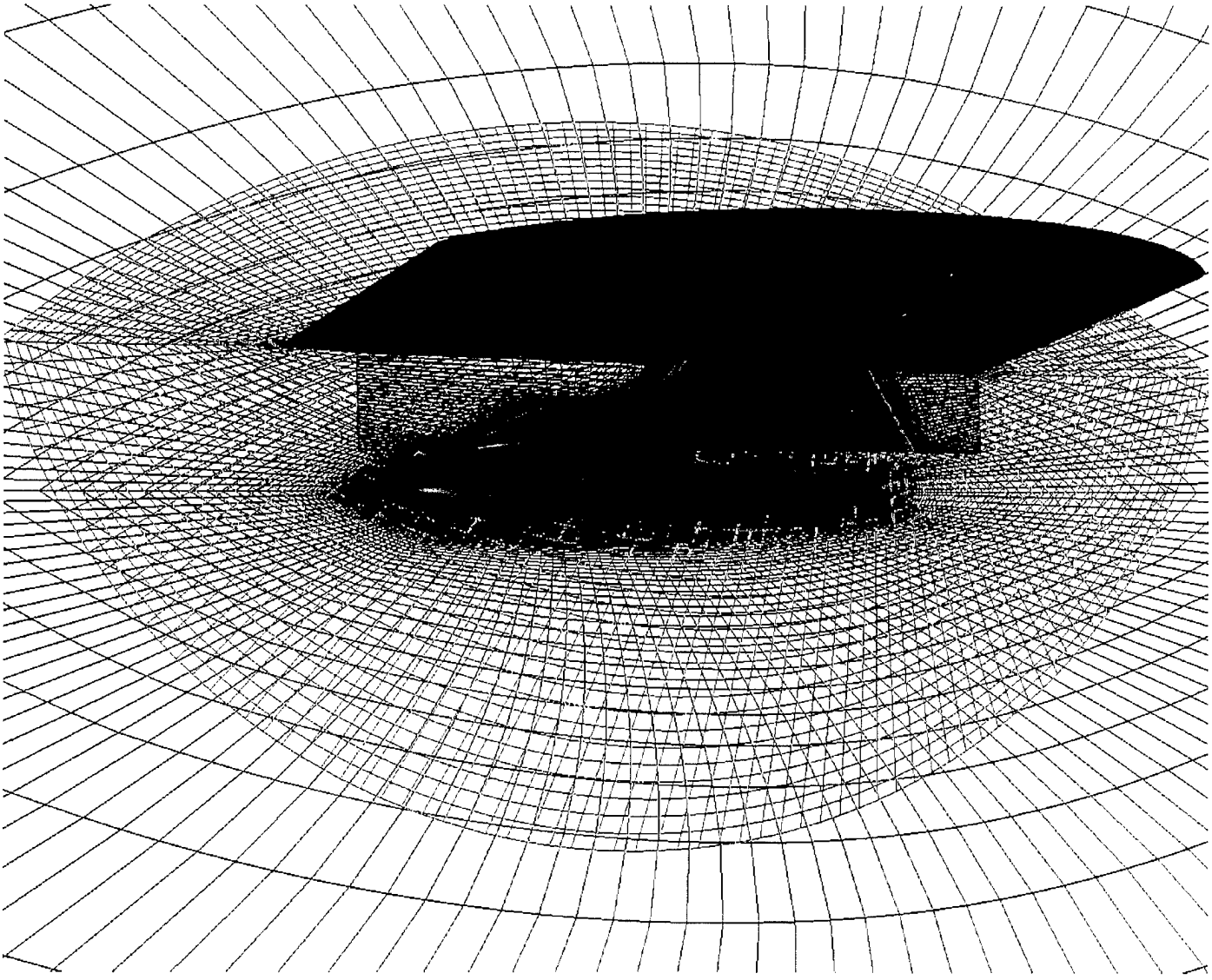


Figure 24. Multi-grid configuration, CATM, pylon and wing.

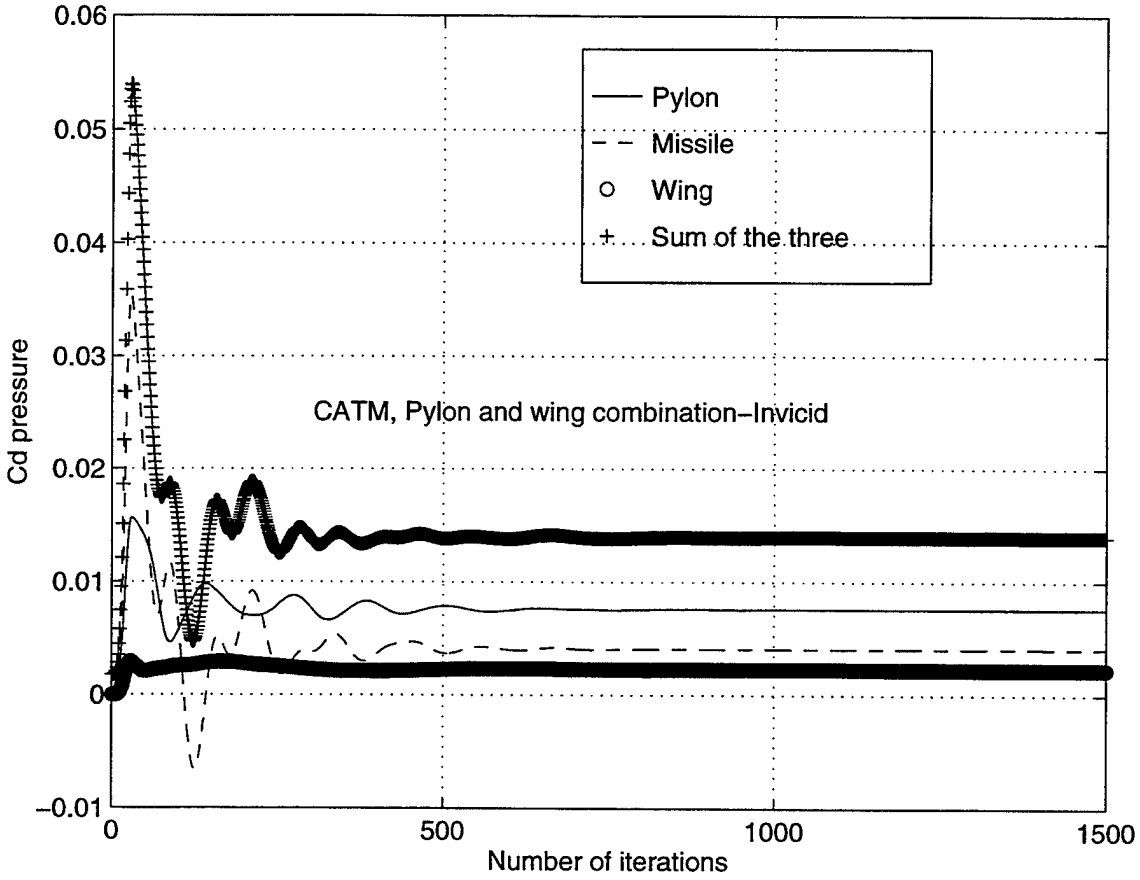


Figure 25. Cd pressure development for the CATM, pylon and wing, inviscid flow.

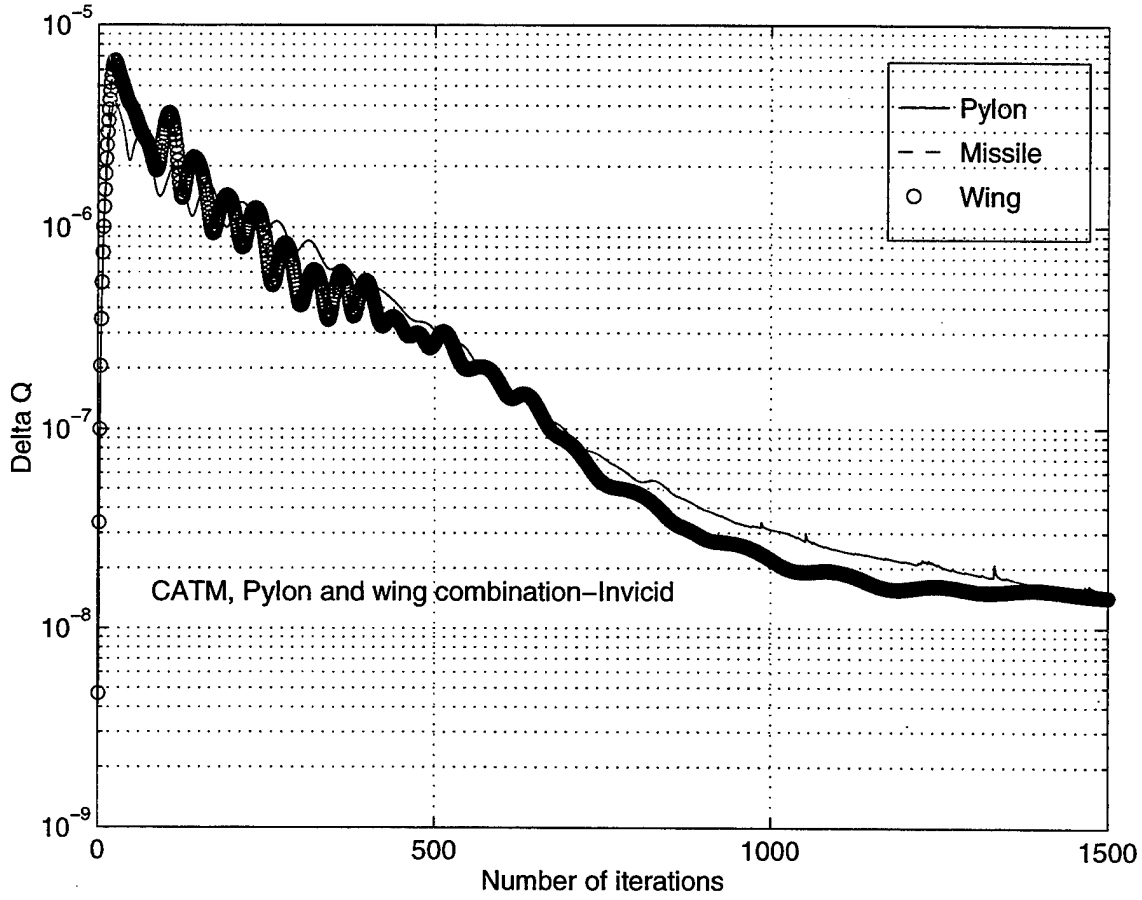


Figure 26. Residuals development for the CATM, pylon and wing, inviscid flow.

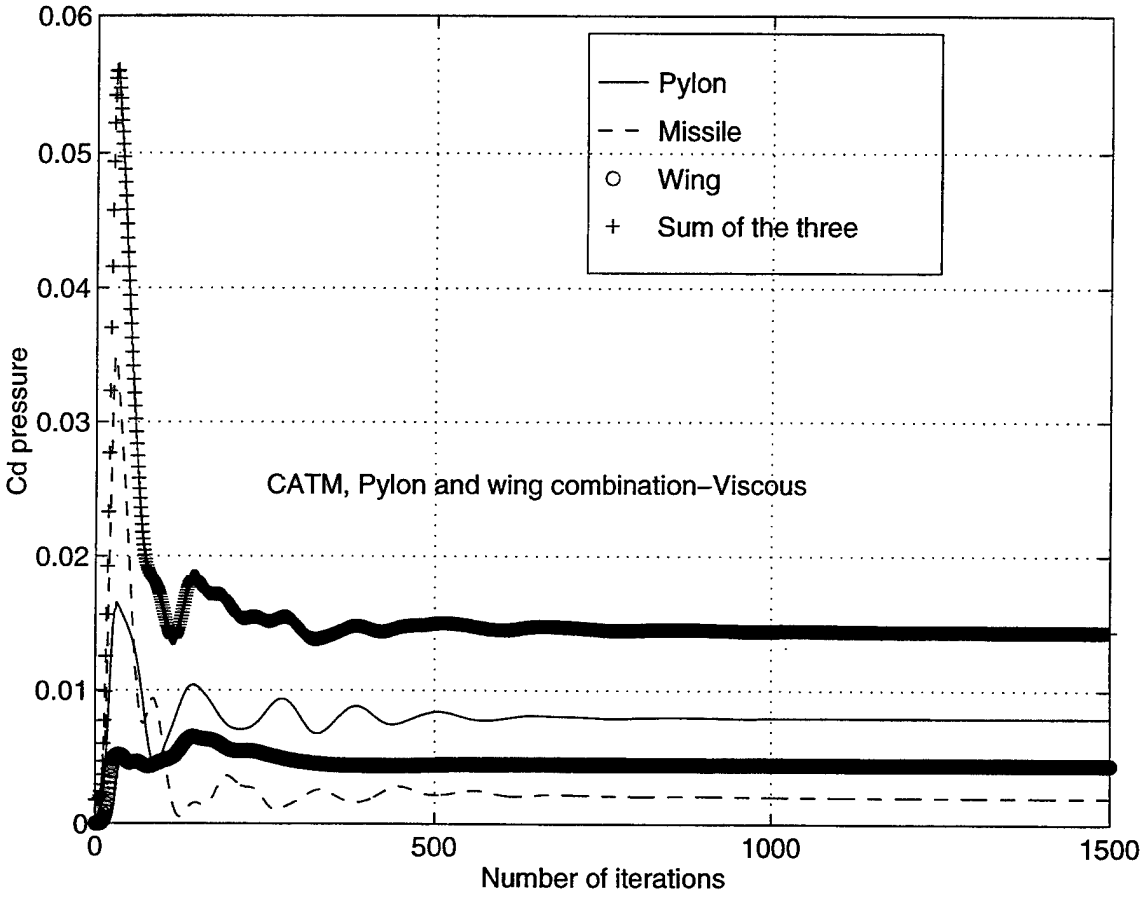


Figure 27. Cd pressure development for the CATM, pylon and wing, viscous flow.

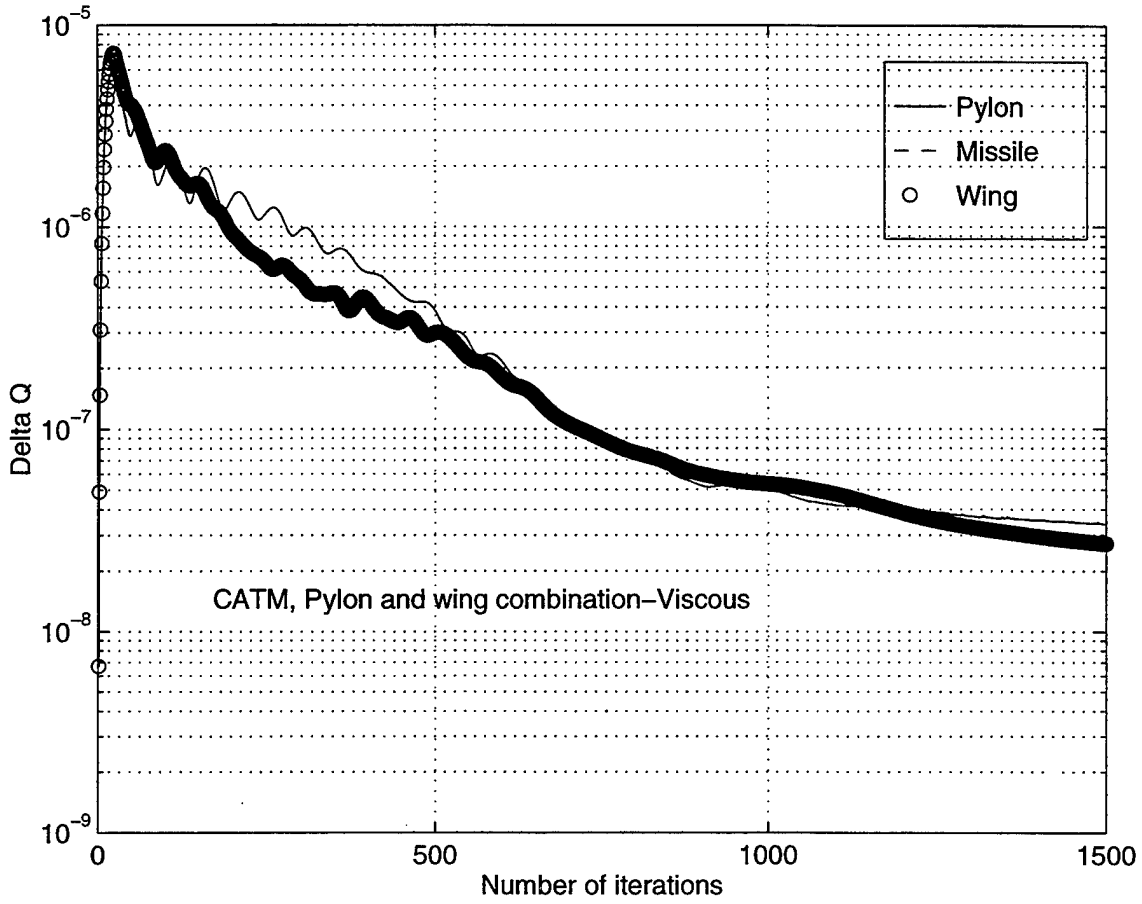


Figure 28. Residuals development for the CATM, pylon and wing, viscous flow.

It can be seen that, as was found in the single grid case, the difference between the viscous and inviscid solution is not a dramatic one. Typical Mach and pressure distributions are given in Figs. 29 through 32.

In the following sections, the results are presented for the multi-grid analysis as a tool for the investigation of the Area-Rule.

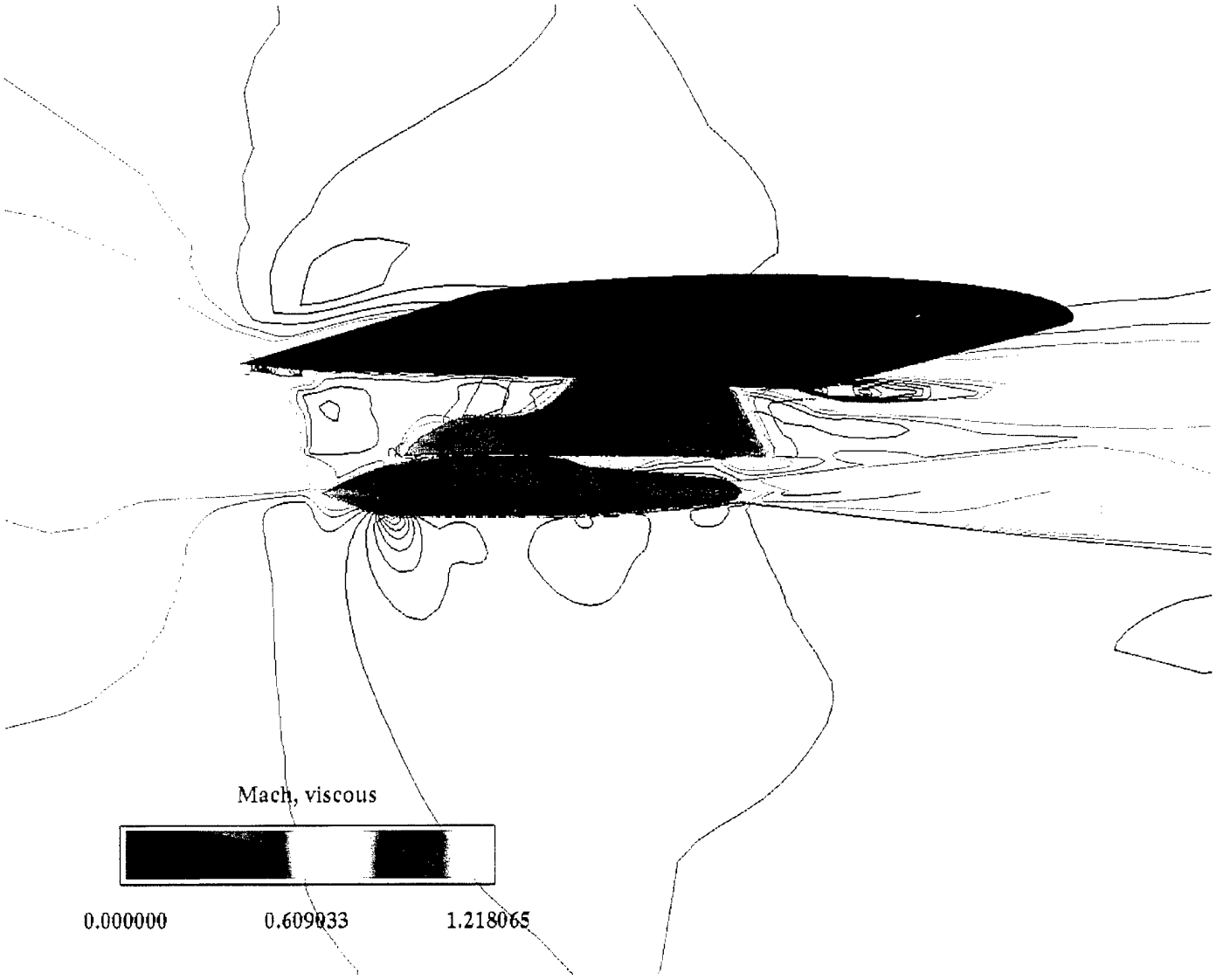


Figure 29. Mach number distribution for a multi-grid case - NS solution.

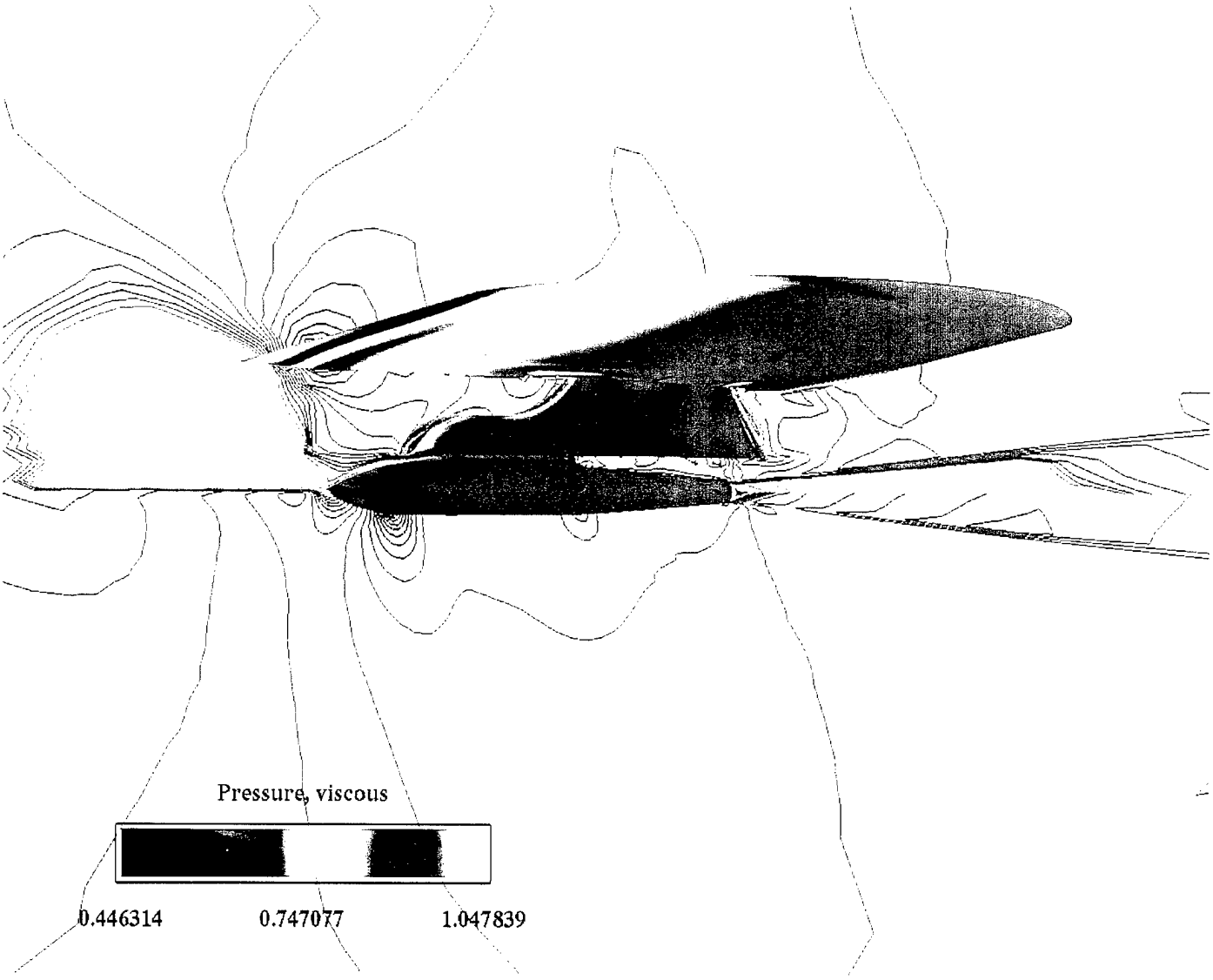


Figure 30. Pressure distribution for a multi-grid case - NS solution.

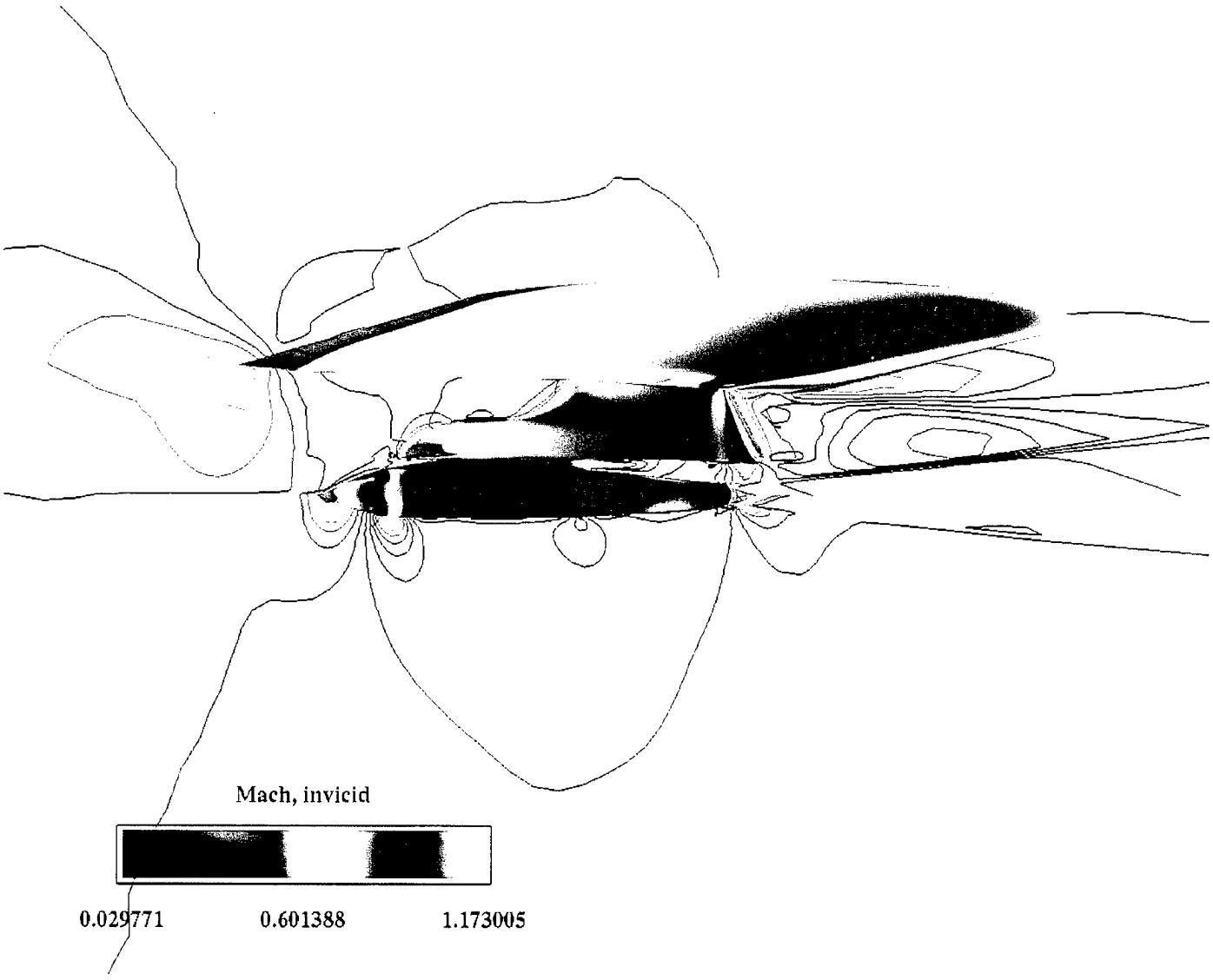


Figure 31. Mach number distribution for a multi-grid case - Euler solution.

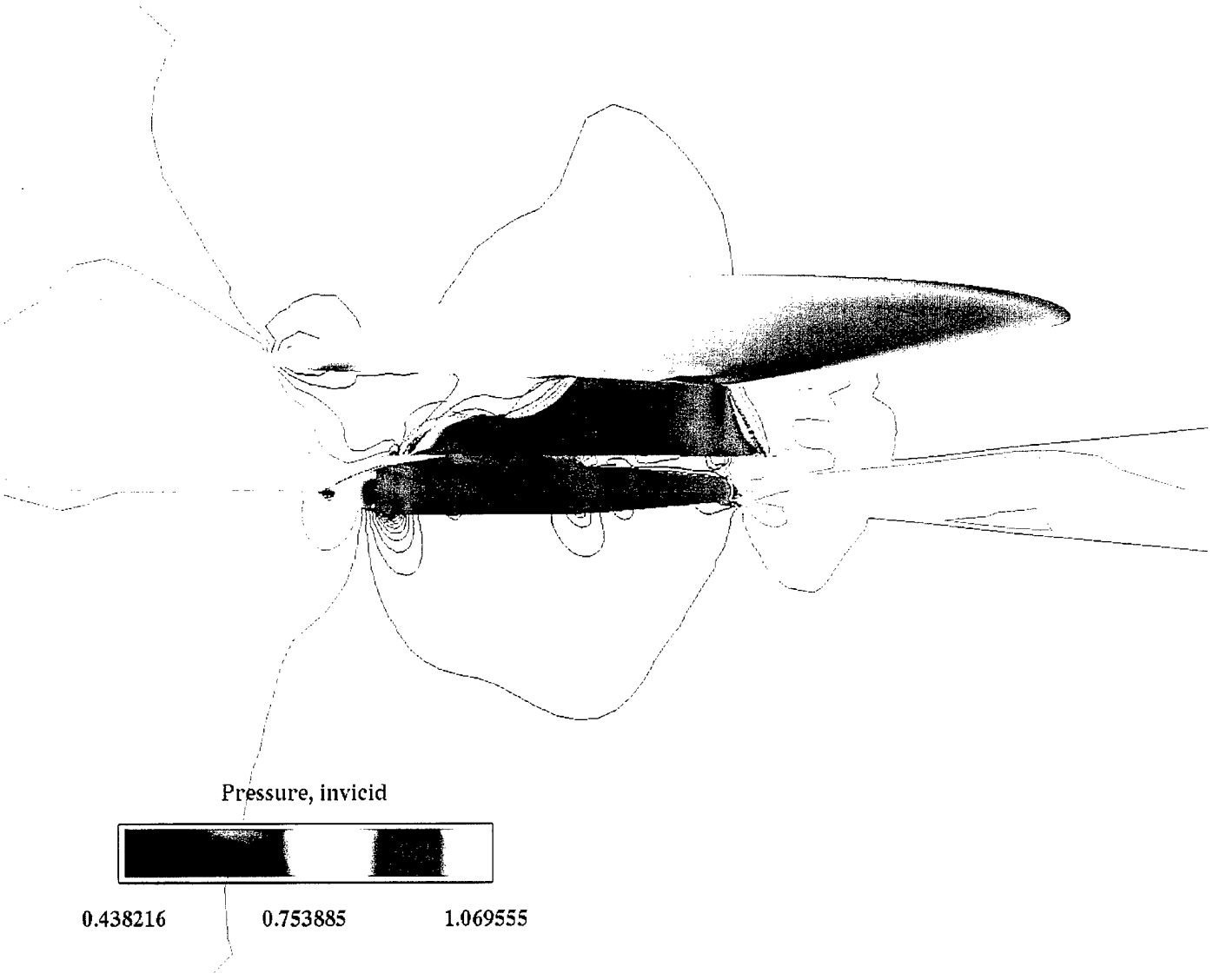


Figure 32. Pressure distribution for a multi-grid case - Euler solution.

D. ORPHAN POINTS EXISTENCE SENSITIVITY

1. General

As mentioned previously, the appearance of ORPHAN POINTS is one of the main obstacles in the CFD process. As the number of grids increases, it becomes more difficult to get rid completely of the OPs. In some cases, a "no Orphan Points" situation has been achieved, but the combination of the grid was not physical, as will be elaborated as follows.

2. Hole Definition and Orphan Points

The definition of a hole and extrapolation regime was given in Chapter III. In this research, when three grids merge together (CATM, Pylon and Wing), it was found that in order to get rid from all the Orphan Points, the limits of the grid of the CATM and the Wing to be calculated in the PEGSUS process (I,J,K INCLUDE parameter) can not include the whole range of grid points.

3. The Physical Conflict

The "physical" meaning of the hole definition is as follows: In the absence of Orphan Points, OVERFLOW calculates the flow over the CATM and the Wing as if the flow exists within a narrow portion in the Pylon (which makes the holes in the CATM and the Wing). This is a complete non-physical situation that can be identified by the fact that the flow pattern of the CATM and the Wing remain almost undisturbed on the symmetry plane of the blocks.

Attempts to get rid of all the Orphan Points and still preserve the physical condition of no flow through the Pylon failed. The attempts included, beside the techniques mentioned above, changes in the grids as well.

4. Sensitivity Study

To investigate the physical necessity of the absence of Orphan Points, the PEGSUS input file was modified to prevent a "flow in the Pylon " situation, while ignoring the existence of Orphan Points (which were found in a significant amount). It also was found that the aerodynamic coefficients almost do not change and the flow pattern becomes a "physical" one [Figs. 29 through 32]. However, an optimization and sensitivity study should be done in any particular case to find the optimum situation of Orphan Points existence and to obtain a physical solution.

VII. THE AREA-RULE

A. GENERAL

Early in the 1950's was found that the pressure drag of a specific configuration is highly dependent on the distribution of the cross-section area of the aerodynamic structure along the flow axis [Ref. 1]. The main pioneering work was done by Whitcomb and has been quoted since then in most of the literature.

The benefits of the Area-Rule are significant mainly in the transonic regime. The concept can be demonstrated clearly by the missile and wings configuration (Fig. 33). To optimize the drag, the missile body was narrowed in the central portion and a "bottle neck" configuration was achieved, which compensated for the addition of the cross-section by the wings.

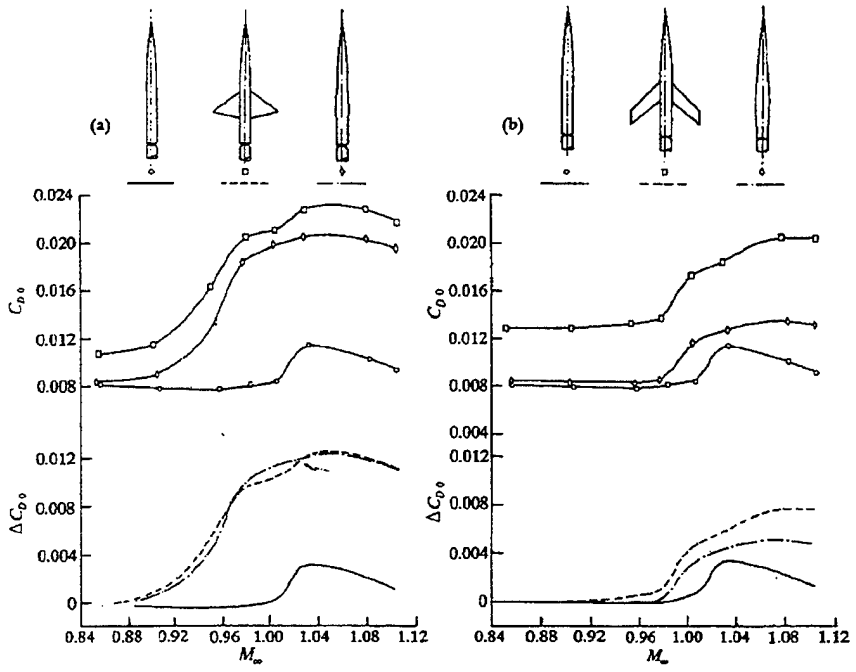


Figure 33. The use of the Area-Rule to reduce drag [Ref.14].

Another implementation of the rule was the increase of the cross-section area of the Boeing 747 passenger airplane at the front end. The central portion of the airplane, where the wings are mounted, was narrowed to achieve a smoother distribution of the cross-section area.

However, the topic was mostly analyzed qualitatively or using parametric analysis. In this research an attempt was made to investigate and verify qualitatively the rule with CFD tools.

A previous research at NPS, dealing with afterbody shape optimization [Ref. 15] found a strong dependence of the drag on the afterbody smoothness. The essence of the research is presented in Fig. 34 where it can be seen that, as the afterbody cross-section area develops in a smoother manner, the transonic drag decreases.

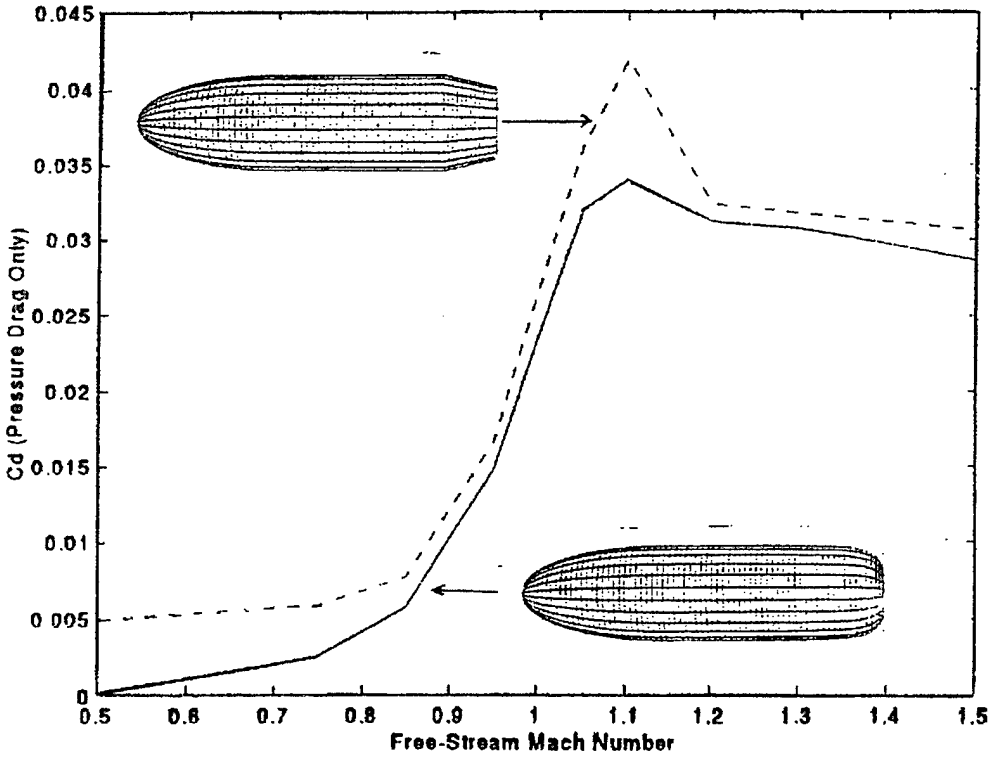


Figure 34. Drag Coefficient for Euler solution [Ref. 15].

Within all the written sources that were investigated in this research, the approach to the Area-Rule analysis was solely qualitative. It was obvious that the smoothness of the cross-section distribution influences the drag directly and the smoother the distribution, the less the drag. Experimental results prove this theory. For example, in the technical note done by Lindsey in 1954 [Ref. 16] decreasing the height of the "bump" in the cross-section distribution plot created by the wing decreased the total drag of the body/wing combination.

However, a quantitative or analytical analysis was not found. The main dilemma was the term "smoothness". This subjective term can be interpreted differently by different individuals. Hence, in this research, with the use of powerful computers and the CFD process, the Area-Rule was modeled for quantitative analysis.

B. THE AREA RULE IMPLEMENTATION

1. General

Originally, the main application of the Area-Rule was to combine slender bodies and wings to smooth the cross-section area distribution of the combination of the aerodynamic structures. In this research, the combination was different, the relative location of the CATM body and the missile was investigated to find whether changing in the cross-section distribution by translating the missile axial location backward and forward, relative to the pylon (Fig. 35), which can optimize the total drag of the configuration.

Aligned Configuration

Forward Configuration

Backward Configuration

Figure 35. Three basic models of CATM/pylon combination to investigate the Area-Rule.

2. Method

The influence of the cross-section area distribution was investigated by creating several combinations of pylon and CATM, of which each combination is different according to the location of the CATM relative to the pylon.

A simple Matlab code was generated [Appendix 6], which loads the grid file as generated by GRIDGEN (after a few modifications such as deleting the "1" and the grid

size and completing the file to a full matrix by adding the appropriate zeros on the bottom of the file in order to be read by the code as a valid matrix).

The code restriction is that the investigated shape must be grided geometry so that there will be constant "I" domains. In the case of distribution along the X axis (or constant "J", "K" in the case of distribution along the Y,Z, axis respectively).

For this reason the pylon domain was hard to analyze, unlike the CATM itself where an exact distribution was found.

The code calculates, for each cross-section, the cross-section area by integrating the area under the curve using the trapezoidal rule.

It should be mentioned that each grid, even if it fulfills the restriction mentioned above, has to be carefully checked before running the code, since the structure of the grid file depends on the direction of the grid and the way the grid was created by GRIDGEN and modified by GRIDED.

As a starting point of work, the following cases were analyzed:

- Aligned case, i.e., the original relative pylon/CATM location.
- Forward case, i.e., the CATM moved forward by 20" (-20").
- Backward case, i.e., the CATM moved backward by 20" (+20").

A distribution cross-section area is plotted in Fig. 36 and the three cases were run in OVERFLOW with and without a wing above. Although the smoothness of the cross-section distribution was improved in the backward case, there is a very sharp apex that causes a rapid change in the derivative. Hence a further optimization was done and the "-7.5 in." was found to be the best in the aspect of "smoothness".

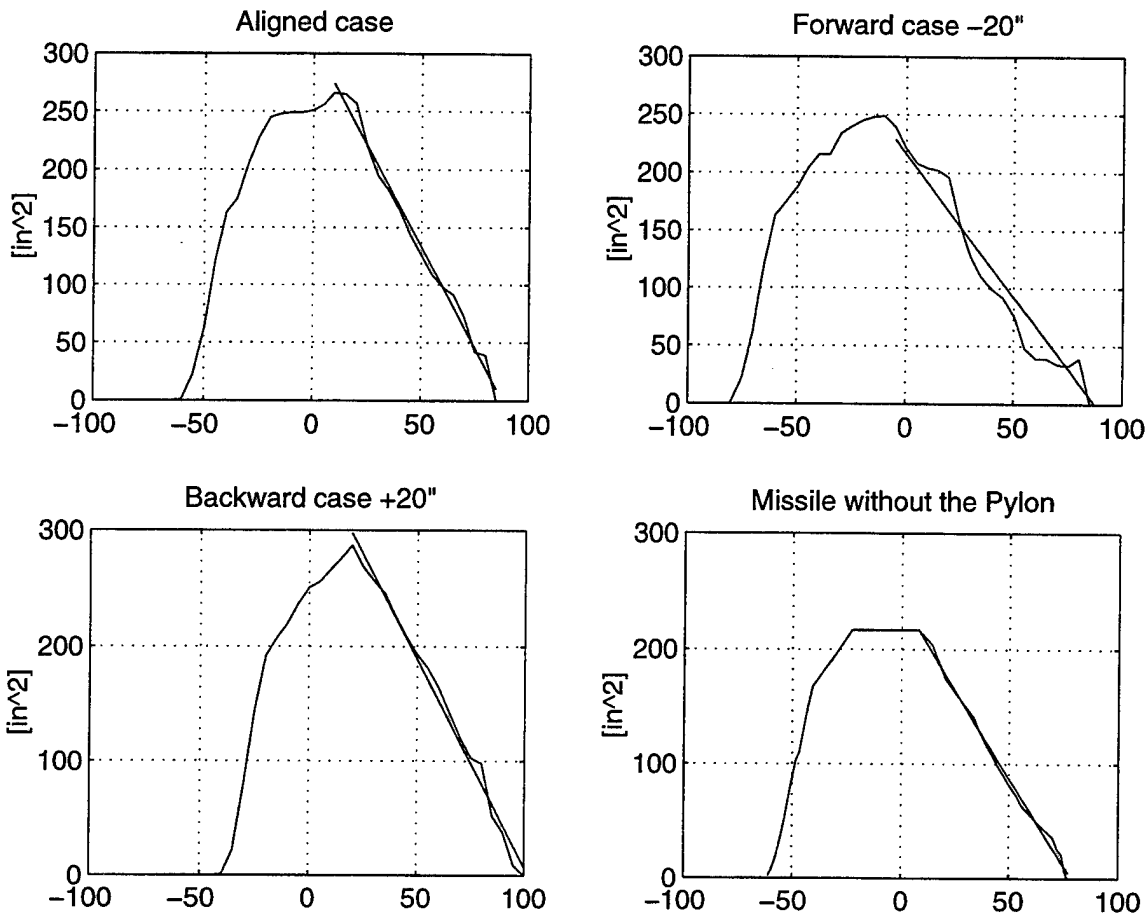


Figure 36. Cross-section area distribution for CATM/pylon combination.

C. INTERFERENCE DRAG

Since we deal with a combination of a few aerodynamic configurations, we must consider the interference drag that occurs between the bodies in conjunction with the Area-Rule.

One of the ways to control the interference drag is by "area ruling" the specific combination, i.e., by adjusting the cross-section area distribution to be as smooth as possible and, according to our research results, to control the slope of the afterbody.

Henfer and Bushnell give in Ref. 17 an example of a wing and nacelle, which is very similar to our case of the CATM and the pylon combination.

The essence of the interference/area ruling can be presented in the charts shown in Fig. 37.

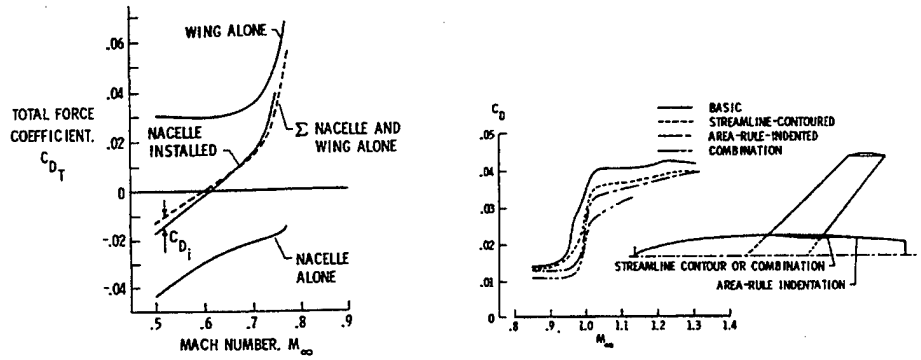


Figure 37. Wing/nacelle example of drag reduction and interference drag [Ref. 17].

The results of this research described in Fig. 38, show a relatively high interference drag contribution. The drag of the CATM without the Pylon, i.e., without interference effects, is much lower then the drag of the various combinations of CATM/Pylon.

D. AREA-RULE RESULTS

1. No minimum value for drag was found for a specific cross-section distribution.

2. The drag increases as the CATM location moves toward the backward of the pylon, and decreases as the CATM location moves forward, until it gets to the value of free-stream conditions. This can be explained by the lack of interference drag as the CATM moves far away from the pylon.

3. Since no special dependence was found for the cross-section area distribution, other "rules" were investigated, and it was found quite clearly that the drag of the CATM decreases as the back slope of the area distribution graph decreases.

Using the code described above, several relative CATM/Pylon location were investigated (aligned case, +20", -20", -10". -7.5" and -5"). For each case, the area cross-section distribution was evaluated and, using linear regression, the slope of the back part of the graph was evaluated. The results are presented in Fig. 38 and include the reference of the free-flight case.

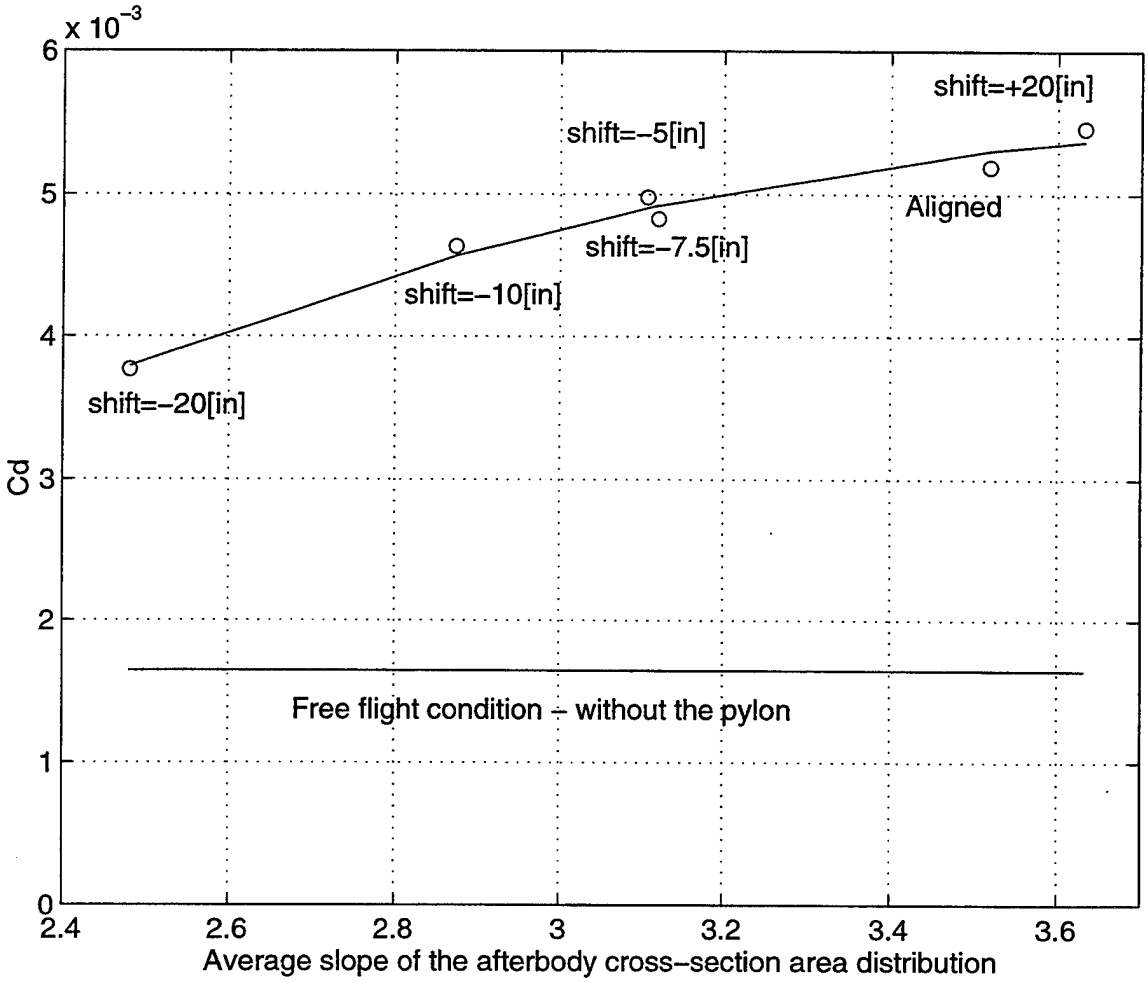


Figure 38. Cd as a function of the average slope of the afterbody cross-section area distribution.

VIII. THE EQUIVALENCE RULE

A. GENERAL

The Area-Rule, can be considered an extension of another significant rule, the Equivalence-Rule [Ref. 14].

In essence, the equivalence rule is a tool for treating relatively complicated aerodynamic configurations by comparing the investigated configuration to a simpler one that has a more trivial solution, for example, a slender body of revolution.

Particularly in the transonic regime where the emphasis of this research lies, one can reduce the linearized, general compressible flow equation [Ref. 14]:

$$\left(1-M^2\right)\Phi_{xx}+\Phi_{yy}+\Phi_{zz}=0 \qquad (22)$$

to the simple Laplace equation:

$$\Phi_{yy}+\Phi_{zz}=0 \qquad (23)$$

and apply the perturbation theory on the simple model.

In subsonic and transonic flows, the far-field flow structure depends on the body **shape** and **volume**, but in the transonic regime the far field depends on the **volume** only. On the other hand, in transonic flows close to Mach 1, the waves move out in a perpendicular direction to the flow. Hence we can deduce that, for the transonic regime, the far-field flow depends on some configuration, not necessarily the actual, investigated one, that has the same cross-section area distribution as the original geometry.

Naturally, we would prefer to choose a simple body of revolution that has the same cross-section area distribution and is easy to deal with in a computational point of view.

B. SMALL PERTURBATION THEORY

The assumption that the equivalence theory is based on the small perturbation theory [Ref. 14] leads to the relatively simple irrotational flow. Its velocity would satisfy the gradient of the potential $\Phi(x,y,z)$ which can be represented, in a zero angle of attack as:

$$\Phi(x,y,z) = U_{\infty}(x) + \tilde{\Phi}(x,y,z) \qquad (24)$$

where $\tilde{\Phi}(x,y,z)$ satisfies the following differential equation:

$$(1-M_{\infty}^{2})\tilde{\Phi}_{xx} + \tilde{\Phi}_{yy} + \tilde{\Phi}_{zz} = M_{\infty}^{2}\left[(\gamma+1)/U_{\infty}\right]\tilde{\Phi}_{x}\tilde{\Phi}_{xx} \qquad (25)$$

C. THE AREA-RULE WITH THE EQUIVALENCE BACKGROUND

Heinrich [Ref. 14] discussed the equivalence rule in detail and derived the exact expressions for the solutions of the basic perturbations equation for the transonic regime, where:

$$\alpha_{p} = M_{\infty}^{2}\frac{\gamma+1}{U}\Phi_{xx} \qquad (26)$$

which is similar to the heat equation and is its solution.

The solution for the transonic regime will be:

$$\frac{u}{U}=\frac{S''(x)}{4\pi}\ln\frac{\alpha_p Se^c}{4\pi x}+\frac{1}{4\pi}\int_0^x\frac{S''(x)-S''(\xi)}{x-\xi}d\xi \qquad (27)$$

and for the equivalent body:

$$\frac{u}{U}=\frac{S''(x)}{4\pi}\ln\frac{S}{\pi} \qquad (28)$$

The smoothness of the cross-section area can be well represented by the behavior of the second derivative of the cross-section area function, $S(x)=\pi r_b^2$. A sudden change in the cross-section area function will increase abruptly the value of $S''(x)$ and hence increase both the drag of the equivalent and the original body.

The subtraction in the last equation represents the contribution of the Cp which originated in the cross-section area distribution along the body axis.

D. SLENDER BODY THEORY

The Area-Rule can be most useful when it is applied on a simple configuration as a slender body. Finding a slender body with a cross-section distribution that is similar to the investigated configuration, can make the transonic drag analysis much simpler.

The slender body theory [Ref. 14] states that the flow potential around a slender body satisfies the following:

$$\Phi=\Phi_2+g(x) \qquad (29)$$

where Φ_2 is the solution for the 2D degenerated equation (Laplace's form) and g(x) is an addition function that depends on the free stream condition and the cross-section area distribution .

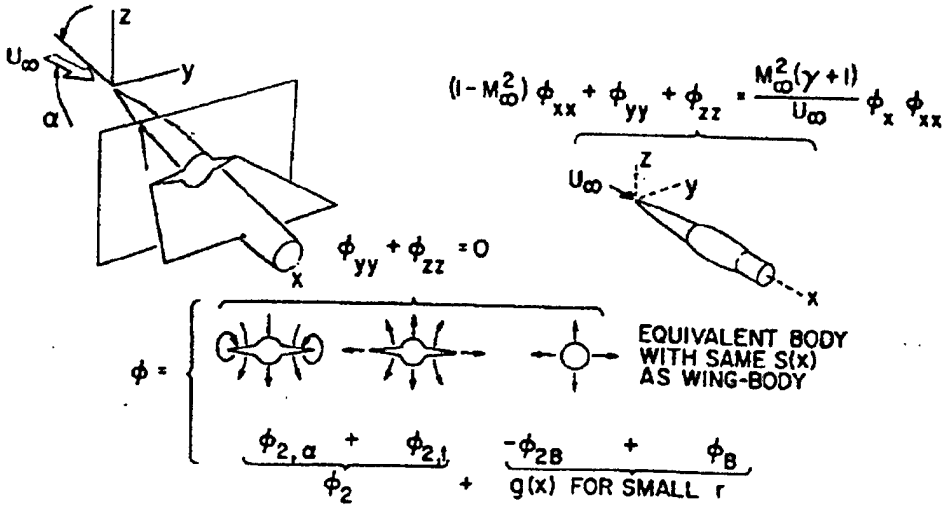


Figure 39. Illustration of the transonic equivalence rule [Ref. 14].

The pressure distribution on a slender body of revolution is as follows [Ref. 14 and 17]:

$$C_P = \frac{p-p_\infty}{\rho_\infty U_\infty^{\,2}/2} = -\frac{2}{U_\infty}\Phi_x - \frac{1}{U_\infty}\left(\Phi_y^2 + \Phi_z^2\right) =$$

$$\frac{2}{\gamma M_\infty^2}\left\{\left[1+\frac{\gamma-1}{2}M_\infty^2\left(1-\frac{(U_\infty+u)^2+v^2+w^2}{U_\infty}\right)\right]^{\frac{\gamma}{\gamma-1}} - 1\right\} \qquad (30)$$

After further simplification, assuming the square of the velocities in the y, z are much smaller then in the x direction, and after applying binomial expansion, we get:

$$C_P = -\frac{2u}{U_\infty} - (1 - M_\infty^2)\frac{u^2}{U_\infty^2} - \frac{v^2 + w^2}{U_\infty^2} \qquad (31)$$

Obviously, in the transonic regime, the x term can be neglected and the following expression for the pressure coefficient can be obtained:

$$C_{P_{Transonic}} = -\frac{2u}{U_\infty} - \frac{v^2 + w^2}{U_\infty^2} \qquad (32)$$

From here, the total forces on a slender body can be found and, using the combination of the area rule and the equivalence rule, the forces on the investigated configuration can be evaluated.

E. THE TRANSONIC SIMILARITY RULE

1. The powerful ability of the similarity rule can be found specifically in the transonic regime where the flow over the body reaches the M = 1 limit. The main obstacle is to measure and calculate the aerodynamic coefficients. The similarity rule makes it possible to use data recorded for subsonic and transonic regimes, and interpolate it for the problematic transonic regime.

2. In the transonic regime, when the first shocks are being built, one can assume that the gap from M = 1 is identical before and after the shock [Ref. 14]. If we denote I, II as the conditions before and after the shock, we will get:

$$M_I - 1 = 1 - M_{II} \qquad (33)$$

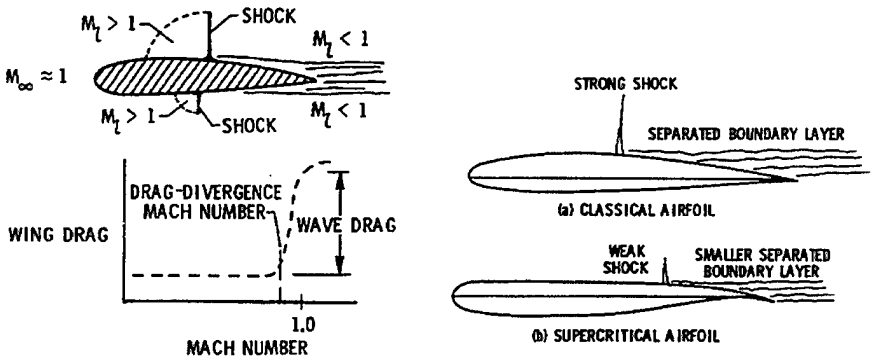


Figure 40. Transonic wave drag [Ref. 17].

The pressure coefficient derivative with respect to the free-stream Mach number, can be found from the expression for Cp. Assuming the slender body and small perturbations theory, we get:

$$\left(\frac{dC_p}{dM_\infty}\right)_{M\infty=1} = \frac{4}{\gamma+1}\left(1-\frac{1}{2}\left(C_p\right)_{M_\infty=1}\right) \approx \left(\frac{4}{\gamma+1}\right) \quad \text{(for slender body)} \qquad (34)$$

The derivation for the drag and lift coefficient is clear, and will not be presented here, although the final implementations can be seen in Fig. 41 [Ref. 14]. For a wedge of thickness h, the drag coefficient at M=1 can be found for various wedge angles and generally, as was found by Spreiter [Ref. 18], as a function of the similarity variable χ.

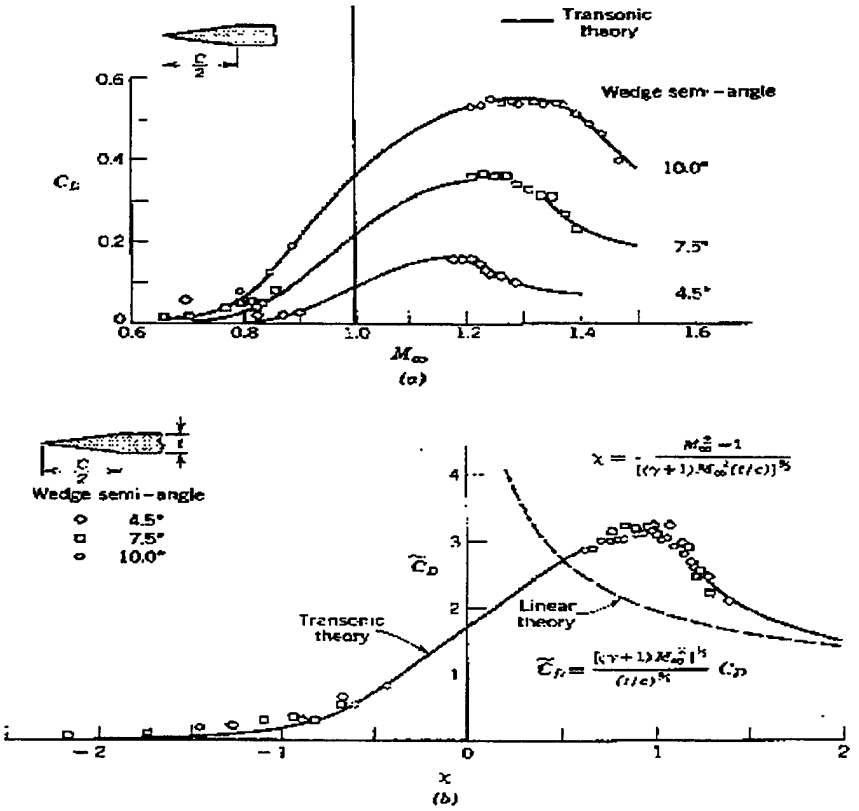


Figure 41. Cd as a function of free stream Mach number - M_∞ and similarity variable-χ [Ref. 14].

F. COMPARISON TO OUR RESULTS

In addition to the smoothness influence as was found by Whitcomb [Ref. 1] and reported by others such as Heinrich [Ref. 14], it was found in this research that the cross-section area distribution influences the drag behavior, not only according to the smoothness of it's development, but according to the slope of its development along the afterbody.

IX. SUMMARY, CONCLUSIONS AND RECOMMENDATIONS

A. GENERAL SUMMARY

1. A CFD solver was applied to a multi-grid configuration and the precise procedure has been documented in detail so as to assist further projects.

2. CFD process was found to be very sensitive to the grid configuration. The grid architecture, including grid line distribution, blocks structure, and grids merging, significantly affects each stage of the process, particularly OVERFLOW.

3. A primary dependence of the drag on afterbody average cross-section area distribution slope was found which can guide aerodynamic shape development for minimum drag. This addition to the traditional Area-Rule can be a powerful tool to accelerate development time.

B. CONCLUSIONS

1. The minimum drag of the CATM mounted on a pylon was found when the CATM is shifted 20" toward flight direction (-20" configuration) and its value is 0.003766. It can be seen (Fig. 38) that the drag coefficient reduces as the CATM gets further from the pylon toward the free flight condition.

2. The drag behavior of the CATM/pylon combination can be explained mainly due to the interference drag caused by the CATM/pylon relative location.

3. Area Rule was found to be dependent on the afterbody cross-section area distribution average slope, and not on the overall smoothness of the cross-section distribution of the CATM/Pylon configuration. The average afterbody cross-section area

distribution correlates well with the drag of the different relative location of the CATM/pylon

4. The results of Euler solutions rather than NS solutions are, in most cases, quite adequate and save the computational efforts.

5. Using CFD, Similarity-Rule can was verified only qualitatively, further work has to be done to analyze the Similarity-Rule quantitatively as was done with the Area-Rule.

6. Orphan Points can be found in the multi-grid approach, but are tolerable as long as their existence does not affect the physical meaning of the solution. In any case, the preferred solution contains no Orphan Points, but, achieving this is not always worth the effort.

7. The use of the NPS Cray services as a main operating system causes significantly delays and increases time response for the results. The optimum and CPU time should be developed economically in order to minimize negative effects, either by using the correct "*.nqs" file or by carefully selecting the critical parameters to achieve rapid convergence.

C. RECOMMENDATIONS

1. To simplify the work in modeling the CATM configuration, and make the results more accurate, a current geometrical data base should be obtained from the contractor (TI) that can be read by the grid generation code in NPS, such as GRIDGEN.

2. The best location to minimize drag is where the CATM is as far forward as possible on the pylon. This result affects structural considerations on the one hand, and

antenna locations (GPS on the-front end of the CATM and communication on the rear-end) on the other. A CATM placed too far forward will cause difficulties for the guided aircraft because of antenna blocking.

3. The basic study of any multi-grid case configuration should start by studying each of the grids separately to assure valid grids that can produce *physical solutions*.

4. When dealing with multi-grid cases, an outer box should be used unless the original, investigated blocks are as broad that the effects from the body do not propagate out to the block boundaries. Therefore, verifying the characteristics of the separated grids, should be done in case an extra block has to be added that will result in significant increases in the computational needs and complexity.

5. Since OVERFLOW also runs on SGI machines, it is recommended to run the initial, non-CPU consuming cases on the SGI machines and to check them directly by FAST (i.e., "q.save" generated by OVERFLOW, which runs on the SGI can be read directly by FAST as a "solution", "unformatted" file, on the other hand, in the Cray case, the "q.save" file has to be converted to a SGI readable format using a "wrq" conversion file).

6. The complexity of the CFD process can be eased by making the codes more user friendly. Recommendations for such changes are given in Appendix E.

7. Since grid generation is one of the most difficult phases in the CFD process, a reliable grid generation tool is required. GRIDGEN is an excellent tool, and it is recommended that the most current version be installed including all the technical support that accompanies it.

8. As mentioned in chapter III-F, the GRIDED version currently installed is not the latest version which has more useful applications. It is recommended to get the updated version, which can help to create simple blocks easily by modifying 2D grids.

9. A further study of a full JSOW including wings and fins is recommended in order to analyze the aerodynamic characteristics of the operational weapon system to be used in other applications such as guidance simulation.

10. Some work should be done to analyze the store-separation behavior of the CATM. Using two codes that were written by Nielson Engineering, SUBSAL and SUBSTR, one can investigate the store trajectory after modeling the store as an equivalent axisymmetric object. The codes can be provided by NAWC China Lake and are ready to use on the SGI machines. This process might complete (partially) the picture by analyzing the CATM separation from the F-18 as a model carrier.

LIST OF REFERENCES

1. Whitcomb, T.R., NASA Report 1273 (1952); also, *Aviation Week and Space Technology*, Sept. 19, 1955 and IAS paper 601 (1956).

2. "US authorized two further JSOW versions", *IDR International Defense Review - Jane's Magazine*, July 1995.

3. Fulghum, David A., "New Wartime Roles Foreseen For JSOW", *Aviation Week and Space Technology*, February 28, 1994.

4. John, D., "NAVY Primed for JSOW critical Design Review", *Aviation Week and Space Technology*, February 27, 1995.

5. Kern, Steven B. and Bruner, Christopher W., "External Carriage Analysis of a Generic Finned - Store on the F-16 using USM3D", AIAA-96-2456-CP.

6. Rom, Josef , "*High Angle of Attack Aerodynamics*", Springer - Verlag,1991.

7. Barton, Bret S., "Application of Multi- Block CFD Techniques to a Missile Geometry", Master of Science in Aeronautical Engineering, Naval Postgraduate School, December 1995.

8. Steinbrenner, John P. and Chawner, John R., "GRIDGEN Release Notes: Version 9.6 (First Production Release)", MDA Engineering, Inc., October 18, 1996.

9. Suhs, N. E. and Tramel, R. W., "*Pegsus 4.0 User's Manual*", Calspan Corporation/AEDC Operations, June 1991.

10. Buning, Peter G., NACA Ames, "*Overflow User's Manual*", Version 1.6aw 12, July 1995.

11. Merkle, Charles L., "Computational Fluid Dynamics of Inviscid and High Reynolds Number Flow", AA4507 notes, Fall September, 1990.

12. Scarry, Michael T., "JSOW CATM Conceptual Weight and Aircraft Design ", Master of Science in Aeronautical Engineering, Naval Postgraduate School, September, 1996.

13. Biblarz, O. and Priyono E., "Transonic Pressure Drag Coefficient for Asymmetric Bodies", *Proceedings, ICAS-94-2.5.2 19th Congress of ICAS*, Anaheim, CA, 1994.

14. Heinrich, J. Ramm, "*Fluid Dynamics for the Study of Transonic Flow*", New York - Oxford, Oxford University, 1990.

15. Priyono, Eddy, "An Investigation of the transonic pressure drag coefficient for axi-symmetric bodies", Master of Science in Aeronautical Engineering, Naval Postgraduate School, March 1994.

16. Lindsey, W.F., "The Flow Past An Unswept And A Swept Wing Body Combination And Their Equivalent Body Of Revolution At Mach Numbers Near 1.0", National Advisory Committee For Aeronautics, Technical Note 3703, 1954.

17. Hefner, Jerry N. and Bushnell, Dennis M., "An Overview of Concepts for Aircraft Drag Reduction", NASA Langgley Research Center, Hampton, Virginia, AGARD-R 654, 1977.

18. Spreiter, John R. and Stahara, Stephan S., "Aerodynamics of Slender Bodies and Wing-Body Combinations at $M_\infty = 1$", Nielsen Engineering & Research Inc., Mountain View , Calif., AIAA V 1 9, No. 9 Sep 1971.

19. Fan, Yue Sang, "An Investigation of The Transonic Drag Coefficient for Axi - Symmetric Bodies", Master of Science in Aeronautical Engineering, Naval Postgraduate School, March 1995.

APPENDIX A. GRID FILES DESCRIPTION.

A. GENERAL

The evolution of this research is reflected by the grids created throughout the work. Since the major obstacles were found in the grid generation and caused by grid structures, we found it very important to keep a detailed description of the grids and their related phenomena, although most of them probably will be erased after the completion of this research.

Generally, each grid was modified after it was created to achieve a uniform configuration. The modification was done by GRIDED and the modified grid was distinguished from the original one. The original grid has a ".grd" suffix, the same grid without the "1" on the top of the file (which indicates for GRIDGEN that it is a one-block grid and has to be eliminated in order to be read by GRIDED), has the "_m.grd" suffix. The same grid after being modified by GRIDED has, in addition a suffix indicating the changes done by GRIDED, e.g., "_4_100.grd" means the grid changed J, L indices (option "4" in GRIDED) and changed the direction of J index. Finally "uf" suffix indicates that the file is unformatted (and can be used directly for OVERFLOW needs).

The following description includes only the basic configurations.

B. THE GRIDS

1. Test case grids

a. test.grd - 80x39x75

This grid was generated to study the basic concept of OVERFLOW after attempts to get a valid solution for the CATM itself failed because of consistent unexplainable negative values of Cd for subsonic flow and positive for supersonic, as the CATM grid, it is a 80x39x75 grid, symmetric on Z axis. The purpose was to create the simplest grid that won't make any special difficulties due to asymmetry, non slender etc. No good solutions were achieved using this grid, probably because of two reasons.

- Although the grid was symmetric, it was not a body of revolution; therefore relatively sharp corners were formed, including on the plane of symmetry. The thought was that corners were not handled by denser grid lines might cause divergence of the solution.
- The other suspected reason is the distribution of the grid lines from the surface outward, although a geometric distribution function was applied (ΔS Initial -0, final 20). The grid was not dense enough, or the ΔS Initial -0 input created difficulties.

b. test1.grd - 80x77x75

One reason for the failure of the test.grd file may be that the boundary conditions definition was incorrect, especially the symmetric BCs. In order to change the BCs radically and check the theorem, a full size grid was created (not a half configuration).

The grid was not used, since, it creates negative volumes in OVERFLOW with no ability to overcome them.

c. test_densed.grd - 80x39x75

The same as test.grd but with a different grid distribution , ΔS Initial -0, final 50. It appeared that even this distribution did not help and the negative Cd continued to appear.

d. Baseball.grd - 80x39x75

Since the test.grd grid was not a body of revolution, a similar grid was created, but this one was a body of revolution. The change was done in order to investigate the "kinks" in the grid, on the symmetry planes in the solution. No improvement was found, and the same negative Cd occurred, which indicates that these "kinks" have no significant influence on the solution.

e. bball_small.grd - 80x39x75

Looking at a few sample cases, such as the Buning sample case and the Barton thesis case, one can see that the width of the blocks is narrow (relative to our blocks, which were made very broad in order to prevent the influence of the block boundaries and reflections). The narrow blocks were used because the cases were multiblock, and on the top of these blocks was an "outerbox" to calculate the far field conditions. However, investigating the grids as a monoblock case, one can find a valid solution. Therefore, a very narrow block was created while the grid size remained the same.

No good solution was found, but the phenomena of negative cells disappeared, and a stable positive value was found, with the restriction of high residuals values (approximately 10E-01 to 10E-03).

f. bb_dense.grd - 80x39x75

After the origin of the negative Cd was found, a new grid was created, identical to the "baseball.grd" but with a different grid distribution. A very dense grid was created near the body surface. To be more specific, a geometrical distribution function was chosen with a ΔS initial and final of 0.05 and 100, respectively (a distribution of 0.01 - 150 failed due to negative cells in OVERFLOW).

Negative Cd was not found, but the lack of convergence was found.

g. Baseball_90.grd - 80x39x75

To achieve a maximum computability between the test case and the Buning test case, the baseball.grd grid was rotated by 90 degrees, so that the symmetry plane was "Y" instead of "Z". The solution, as expected, was the same.

h. bb_dense_ang.grd. - 80x39x75

In a few projects [Ref. 7 and Ref. 19] and in Buning sample case as well, the bodies investigated were not a half body but slightly more, i.e., approximately 200 degrees were investigated rather then 180 degrees as expected. Therefor a new grid was created out of the basic "baseball_90.grd" grid but in 200 degrees. No different phenomena were found.

The reason for the addition of the angle was not clear, especially since the BCs on the inclined surfaces is a symmetry in the "Z" direction and, once the surface is inclined, the symmetry is not purely in the "Z" axis.

2. JSOW/CATM Grids.

a. Jb_block4.grd - 80 x 39 x 85

This is the basic JSOW block that was created. It is a very narrow block. The diameter of the outer shell is approximately twice the length of the missile itself. Hence, phenomenon of shock reflection occurred and can be seen in Fig. A1. The reflected shocks hit the body near its rear end and create a "ring" that can be seen in Fig. A1

b. Jb_block5.grd - 80 x 39 x 85

Due to the shock reflection phenomenon, the block was significantly increased and, indeed, the reflections disappeared.

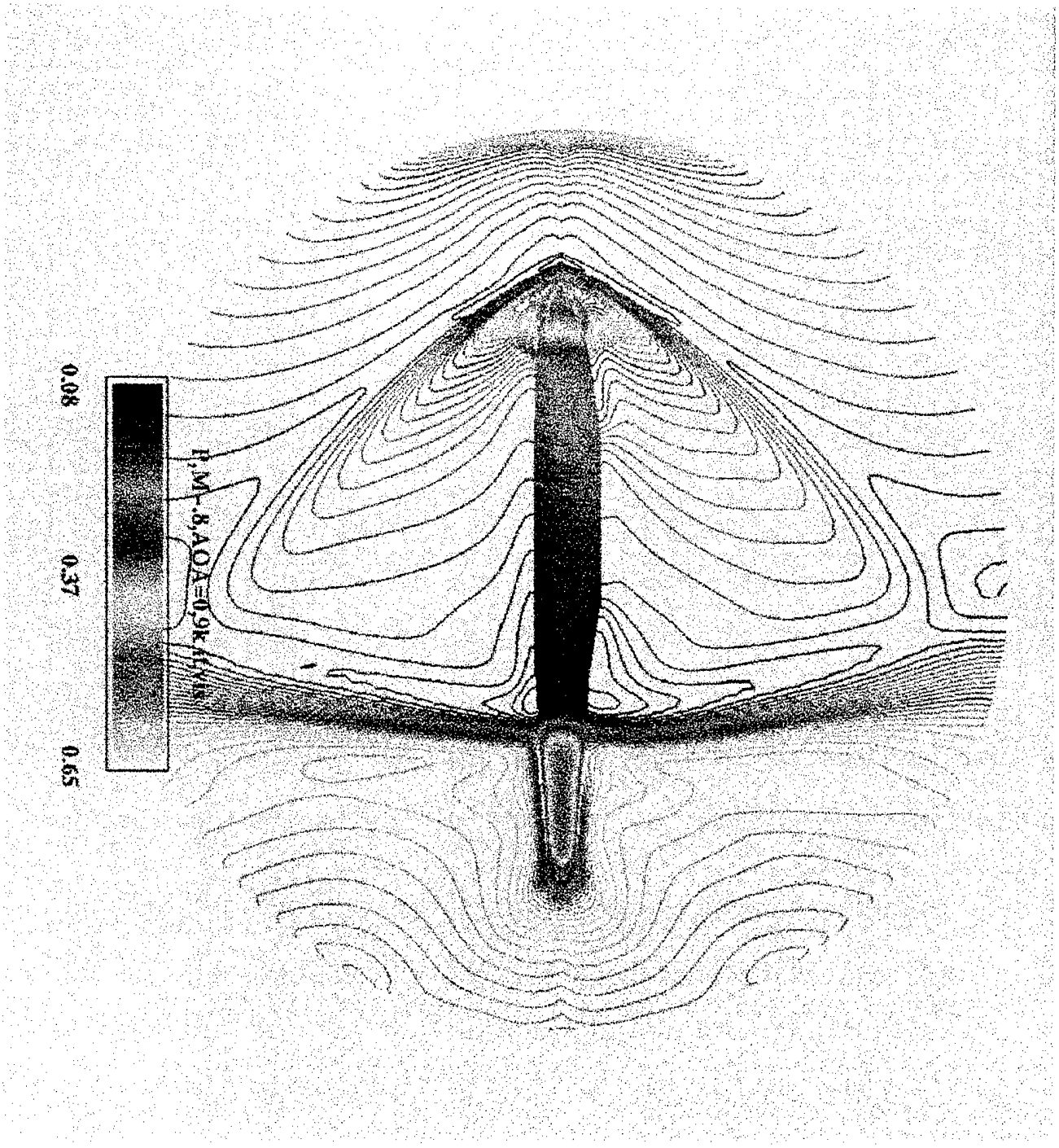


Figure A1. Reflection of shock waves due to too small block size.

c. *Jb_block8.grd - 80 x 39 x 75*

This is the first short version of the CATM after it was modified and its weight was decreased to below 300 lb [Ref. 12]. The grid size decreased a little in the "K" direction. No special changes were made to the grid distribution.

d. *Jb_block9.grd - 80 x 39 x 75*

This is the same grid as the jb_block8.grd, but with a wider block, created to get rid of the "ring" that appears at the rear portion of the missile due to a reflection from the block boundary [Fig. A1]. The "ring" phenomena disappeared, but, the solution gave negative Cd for the subsonic regime (Mach number of 0.5 to 0.8) and a positive Cd in the supersonic regime (a Mach number of 1.2).

e. *Jb_block10.grd - 80 x 39 x 75*

This is a very wide block (radius to body length ratio of about 1500), was created since it was suspected that the previous block was not broad enough (approximately 3 to 4 times the body length) and there still is an influence of the outer shell boundary on the solution. It was found that it is not true, and the block does not have to be so wide, especially in a multiblock grid. The block gave consistently negative Cd.

f. *Jb_block11.grd - 80 x 39 x 75*

This is a block that was created after it was found that the dense grid near the surface is more important than was initially thought. The distribution function was set to "geometric" with an initial and final spacing of 0.05, 150. The negative Cd phenomena did not occur.

The grid was modified again and an extra plane was added to the J domains on the other side of the symmetric plane (plane Z=0). The addition of the plane was done by using the GRIDED feature to add another domain. The purpose of the extra plane is to enable OVERFLOW to imply the BCs on the symmetry plane in so that there will be "sources " of information from both sides of the plane to get by averaging the information from the middle (symmetry) plane.

h. Jb_block12.grd - 80 x 39 x 75

Before the "extra" plane was added to the previous block (Jb_block11.grd), an attempt was made to run a case with a block that was extended to more then 180 degrees. Obviously it was a failure, since only one surface had to be added (as described above) and, in this block, more then one grid was added.

i. Jb_block13.grd - 80 x 39 x 75

Since the Jb_block11.grd block was oriented with the Z axis as the symmetry axis, the definition of α,β was disturbed; hence, an identical block was created that was rotated by 90 degrees relative to Jb_block11.grd in order to verify the definition of the directions of α,β.

Since all the research was done using the Jb_block11.grd block, and the orientation of the pylon and wing blocks was similar to the Jb_block11.grd, the Jb_block13.grd block was used only to verify the directions of α,β.

3. Pylon Grids

The Pylon grid originally was generated as the JSOW/CATM from a set of drawings. Since the main reason for using the Pylon is to investigate the Area Rule rather

than to analyze its aerodynamic behavior, it is not critical that the geometry be identical to the reality.

The shape of the Pylon was given as contour lines only; the details of the geometry was unknown and assumptions had to be made. Many Pylon grids were created; however, unlike the JSOW/CATM case, this block was not a problematic one and through most of the research only one block ("p_densed.grd") was used. The block was modified in several cases by applying the elliptical solver in GRIDGEN to prevent Negative Volumes (p_dense_elip5.grd - 5 iterations of elliptical solver and p_dense_elip8.grd - 8 iterations of elliptical solver).

In the research phase, where Orphan Points were found when a combination between the CATM and the Pylon grids was done, a new block was created - ("p_den_ang.grd") which had a different block structure than the original. If the original block (Fig. 24) was built as half of cylinder, then in the "angle" block, the upper and lower domains were tilted 45 degrees to "open" the block. The reason for doing so was to create grid lines that will spread at an angle and will be more parallel to the grid lines of the CATM. Thus the grid cells of the Pylon far from the Pylon surface will be similar in size to the CATM grid cells in the same area and, the information from the Pylon to the CATM would be propagate in a better way. The modification helped slightly to get rid of part of the OPs; however, the block was not used further because other methods, as described previously, were applied.

4. Wing Grid

The wing grid was created to investigate the sensitivity of the flow pattern around the CATM and under a wing. Since no accurate information about the geometry of a F-18 wing was available, a general wing grid was created, "w_block1_m.grd. The block was

similar to the CATM block, an "O" shape block with two stings and two symmetry domains.

5. Outer Box

As was mentioned previously, the outer box was created but not used much since the CATM block was broad enough that the influence of the body did not propagate to the block boundaries.

The block grid, "outbox_b1.grd",was a simple cubic shape block, in order to save grid points and CPU, the distribution of the grid points was set so that the grid was dense around the CATM / Pylon and the grid became coarser toward the outer boundary of the outer box.

APPENDIX B. FILES OF INTEREST IN THE CFD PROCESS

A. GENERAL

This Appendix includes several files: executables and input files that are being used in the CFD process and can be used as an example for future research.

B. EXECUTABLE FILES:

- xa2b.f
- xb2a.f
- rmg2peg.f
- merge41.f

C. INPUT FILES

- OVERFLOW input file sample.
- PEGSUS input file sample.

D. INFORMATION FILES

- ja_sum.out

```
c-------------------------------------------------
c    xa2b.f - ascii to binary converting file
c-------------------------------------------------
c       this program will read ascii data files and convert them to
c       binary files for plotting in iris.
c       convert from formatted to unformated
c        grid.for to grid.in
c-----------------------------------
      DIMENSION x(75,41,80),y(75,41,80),z(75,41,80)
      OPEN(unit=12,file='grid.for',status='unknown')
      OPEN(unit=14,file='grid.in',status='new',form='unformatted')
      READ(12,*) IDIM,JDIM,KDIM
      READ(12,*) (((X(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM),
     c           (((Y(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM),
     c           (((Z(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM)
      WRITE(14) IDIM,JDIM,KDIM
      WRITE(14) (((X(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM),
     #           (((Y(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM),
     #            (((Z(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM)
      STOP
      END
```

```
c-----------------------------------------------
c     xb2a.f - binary to ascii converting file
c-----------------------------------------------
c        this program will read binary data files and convert them to
c        ascii files for plotting in iris.
c        convert from unformatted to formated
c         grid.in to grid.fmt
c-------------------------------------
      DIMENSION x(75,41,80),y(75,41,80),z(75,41,80)
      OPEN(unit=12,file='grid.fmt',status='unknown')
      OPEN(unit=14,file='grid.in',status='unknown',form='unformatted')
      READ(14) IDIM,JDIM,KDIM
      READ(14) (((X(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM),
     c          (((Y(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM),
     c          (((Z(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM)
      WRITE(12,*) IDIM,JDIM,KDIM
      WRITE(12,*) (((X(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM),
     &           (((Y(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM),
     &            (((Z(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM)
      STOP
      END
```

```
      program rmgrid
c  Merge multiple PLOT3D grids into a single file to be used in
c     pegasus 3.0
c ****************************************************************
c  ngrid    : # of grids
c  forms    : data file format, binary for iris workstation
c             unformatted for navier
c  fno      : output file name
c  fni      : input file name
c  name     : mesh name to be used in pegasus 3.0
      parameter ( meshes=20, mpts=600000 )
      character*23 fni(meshes),fno
      character*11 forms
      character*40 name(meshes)
      dimension x(mpts*3), jmax(meshes), kmax(meshes), lmax(meshes)
      write(*,6) 'Enter # of grids : '
      read(*,*) ngrid
      if (ngrid.gt.meshes) then
         write(*,*) 'Too many grids.  Change meshes to ', ngrid,
     $        ' or bigger'
         stop
      endif
      write(*,6)
     $     'Enter format of the file (formatted / unformatted) : '
      read(*,5) forms
5     format(a)
6     format(1x,a,$)
      iunit=1
      iunito=2
      do 9 i=1,ngrid
         write(*,7) 'Enter name of input file ',i,' : '
7        format(1x,a,i2,a,$)
         read(*,5) fni(i)
         open(unit=iunit, file=fni(i), status='old',
     $        access='sequential',form=forms)
         read(iunit) jmax(i),kmax(i),lmax(i)
         npt=jmax(i)*kmax(i)*lmax(i)
         if (npt.gt.mpts) then
            write(*,*) 'Too many points.  Change mpts to ', npt,
     $           ' or bigger'
            stop
         endif
         write(*,6) 'Enter mesh name to be used in pegasus : '
         read(*,5) name(i)
         close(iunit)
9     continue
      write(*,6) 'Enter output file name : '
      read(*,5) fno
      open(unit=iunito, file=fno, status='unknown',
     $     access='sequential',form=forms)
      do 10 i=1,ngrid
         open(unit=iunit, file=fni(i), status='old',
     $        access='sequential',form=forms)
         read(iunit) jmax(i),kmax(i),lmax(i)
         npt=jmax(i)*kmax(i)*lmax(i)
         write(*,*) 'Reading grid ',i,jmax(i),kmax(i),lmax(i)
         read(iunit) (x(j),j=1,3*npt)
         write(*,*) 'Writing grid ',i
         write(iunito) name(i)
         write(iunito) jmax(i), kmax(i), lmax(i)
         write(iunito) (x(j),j=1,3*npt)
         close(iunit)
10    continue
      close(iunito)
      stop
      end
```

```
C
C   Program to merge files COMPOUT and IBPLOT from PEGASUS 4.01 to form a
C   PLOT3D 3.5 multiple grid XYZ file with blanking.
C
      PARAMETER (MPTS=600000,MMAX=3*MPTS,MGRID=10)
      DIMENSION X(MMAX),IBLANK(MPTS)
      DIMENSION NI(MGRID),NJ(MGRID),NK(MGRID)
      CHARACTER*80 FILE
      CHARACTER*40 NAME
C
C   Open the two PEGASUS files COMPOUT and IBPLOT.
C
      OPEN(UNIT=1,FILE='COMPOUT',STATUS='OLD',FORM='UNFORMATTED')
      OPEN(UNIT=3,FILE='IBPLOT',STATUS='OLD',FORM='UNFORMATTED')
C
C   Read the various grid dimensions and count the number of grids.
C
      NGRD= 0
   10 CONTINUE
      NGRD= NGRD+1
      READ(1,END=20) NG,NI(NGRD),NJ(NGRD),NK(NGRD),NAME
      NGRID= NGRD
      READ(1)
      NPTS = NI(NGRD)*NJ(NGRD)*NK(NGRD)
      IF (NPTS.GT.MPTS) THEN
         CALL TRIM(NAME,LNAME)
         WRITE(*,12) NGRD,NAME(1:LNAME),NPTS,MPTS
   12    FORMAT(' Error reading COMPOUT: too many points in grid ',I3/
     &         ' ',A/
     &         ' Number of points is ',I7,', dimension is ',I7)
         STOP 'ABORT'
      ENDIF
      GOTO 10
C
C   Get PLOT3D filename.
C
   20 CONTINUE
      WRITE(*,21)
   21 FORMAT(' Enter PLOT3D XYZ filename:  ',$)
      READ(*,22) FILE
   22 FORMAT(A)
C
      OPEN(UNIT=2,FILE=FILE,STATUS='UNKNOWN',FORM='UNFORMATTED')
      REWIND(UNIT=1)
      WRITE(2) NGRID
      WRITE(2) (NI(IG),NJ(IG),NK(IG),IG=1,NGRID)
      DO 30 IG= 1,NGRID
         NPTS = NI(IG)*NJ(IG)*NK(IG)
         LEN  = NPTS*3
         READ(1)  NMESH,NIG,NJG,NKG,NAME
         READ(1)  (X(I),I=1,LEN)
         READ(3)  (IBLANK(I),I=1,NPTS)
         CALL TRIM(NAME,LNAME)
         WRITE(*,23) IG,NAME(1:LNAME),NI(IG),NJ(IG),NK(IG)
   23    FORMAT(' Writing grid ',I3,', (',A,')'/
     &          '                dimensions ',3I7)
         WRITE(2) (X(I),I=1,NPTS*3),(IBLANK(I),I=1,NPTS)
   30    CONTINUE
C
      CLOSE(UNIT=1)
      CLOSE(UNIT=3)
      CLOSE(UNIT=2)
C
      STOP
      END
      SUBROUTINE TRIM(STRING,LSTRIN)
C
C   Return the length of STRING after trailing blanks, nulls, and tabs have
C   been removed.
C
      CHARACTER*(*) STRING
C
      CHARACTER NULL,TAB
C
C   Initialize the null and tab characters.
C
```



```
      NULL= CHAR(0)
      TAB = CHAR(9)
C
C   Loop backwards through the character string and find the last nonblank,
C   nonnull character.
C
      LSTRIN= LEN(STRING)
      DO 10 L= LSTRIN,1,-1
         IF (STRING(L:L).NE.' ' .AND. STRING(L:L).NE.NULL
     C                          .AND. STRING(L:L).NE.TAB) THEN
            LSTRIN= L
            GOTO 20
         ENDIF
   10    CONTINUE
C
C   ALL blank or null or tabs!
C
      LSTRIN= 0
C
   20 CONTINUE
      RETURN
      END
```

```
C                OVERFLOW INPUT FILE - SLENDER BODY & CARTESIAN GRID
verflow.1.in

C
C     By Boaz Pomerantz
C        Date: :Feb, 1997
C
C ******************************************************************
C This is a sample OVERFLOW input file for multi-grid case, viscous
C solution.
C ******************************************************************
 $GLOBAL
      CHIMPA= .T.,  NSTEPS= 1500,  RESTRT= .F.,  NSAVE = 1500,
      NQT   = 0,
      $END
 $FLOINP
      FSMACH= 0.85,  ALPHA = 0.00,  REY   = 8.713E6,  TINF  = 518.42,
      $END
 $VARGAM $END

 $GRDNAM
      NAME = 'PYLON',
 $END
 $NITERS
      $END
 $METPRM
      IRHS  = 0,  ILHS  = 2,  IDISS = 3,
      $END
 $TIMACU
      ITIME = 1,  DT    = 0.01,  CFLMIN= 3,
      $END
 $SMOACU
      ISPEC = 2,  DIS2  = 2.00,  DIS4  = 0.04,
      SMOO  = 0.00,
      $END
 $VISINP
      VISCJ = .T.,  VISCK = .T.,  VISCL = .T.,
      NTURB =  0,
     $END
 $BCINP
      NBC   =  6,
      IBTYP =   5, 32, 13, 13, 32, 32,
      IBDIR =   3, -3,  1, -1,  2, -2,
      JBCS  =   1,  1,  1, -1,  1,  1,
      JBCE  =  -1, -1,  1, -1, -1, -1,
      KBCS  =   1,  1,  1,  1,  1, -1,
      KBCE  =  -1, -1, -1, -1,  1, -1,
      LBCS  =   1, -1,  1,  1,  1,  1,
      LBCE  =   1, -1, -1, -1, -1, -1,
      $END
 $SCEINP
 $END

 $GRDNAM
      NAME = 'MISSILE',
 $END
 $NITERS $END
 $METPRM $END
 $TIMACU $END
 $SMOACU $END
 $VISINP
      VISCJ = .T.,  VISCK = .T.,  VISCL = .T.,
      NTURB = 0,
      $END
 $BCINP
      NBC   =  6,
      IBTYP =  5, 32, 13, 13, 15, 15,
      IBDIR =  3, -3,  2, -2,  1, -1,
      JBCS  =  1,  1,  1,  1,  1, -1,
      JBCE  = -1, -1, -1, -1,  1, -1,
      KBCS  =  1,  1,  1, -1,  1,  1,
      KBCE  = -1, -1,  1, -1, -1, -1,
      LBCS  =  1, -1,  1,  1,  1,  1,
      LBCE  =  1, -1, -1, -1, -1, -1,
      $END
 $SCEINP $END
```



```
 $GRDNAM
      NAME = 'WING',
 $END
 $NITERS $END
 $METPRM $END
 $TIMACU $END
 $SMOACU $END
 $VISINP
      VISCJ = .T.,  VISCK = .T.,  VISCL = .T.,
      NTURB = 0,
      $END
 $BCINP
      NBC   =  6,
      IBTYP =  5, 32, 13, 13, 15, 15,
      IBDIR =  3, -3,  2, -2,  1, -1,
      JBCS  =  1,  1,  1,  1,  1, -1,
      JBCE  = -1, -1, -1, -1,  1, -1,
      KBCS  =  1,  1,  1, -1,  1,  1,
      KBCE  = -1, -1,  1, -1, -1, -1,
      LBCS  =  1, -1,  1,  1,  1,  1,
      LBCE  =  1, -1, -1, -1, -1, -1,
      $END
 $SCEINP $END
```

```
C                    PEGSUS INPUT FILE - MISSILE + PYLON
C
C       By  : Boaz Pomerantz
C       Date: Feb. 1997
C
C ***************************************************************
C ***************************************************************
C                         General Information
C
C  This input file is used with the grid file 'INGRID' by PEGSUS 4.0
C  to create the interpolation stencil used by OVERFLOW. The output files
C  COMPOUT (composite mesh) and IBPLOT (iblanking info) are concatenated
C  in program MERGE41 into 'grid.in'. This file and PEGSUS interpolation
C  information file 'INTOUT' are required by OVERFLOW.
C
C ***************************************************************
C ***************************************************************

 $GLOBAL
   QUALITY = 0.9,0.1,-0.1,

   EPS     = 0.005,
   FRINGE = 1,

   $END


C   Exclude boundary points (via the INCLUDE statement) on the downstream
C   boundary, reflected symmetry planes, or surfaces since these points will
C   be initialise directly by OVERFLOW boundary conditions.
C
C   The X0, Y0 and Z0 are translation coordinates which sometimes are
C   useful to eliminate small numbers of Orphan Points. Rotation is also
C   available.

C   Grid Dimen -  PYLON (62,28,80), MISSILE (75,41,80),
C                 WING (80,41,30)

 $MESH NAME = 'PYLON',
       LINK = 'MISSILE','WING',
       JINCLUDE = 2,61,
       KINCLUDE = 2,27,
       LINCLUDE = 2,79,
       X0 = 0.0,
       Y0 = 0.0,
       Z0 = 0.0,
                                  $END

 $MESH NAME = 'MISSILE',
       LINK = 'PYLON','WING',
       JINCLUDE = 2,74,
       KINCLUDE = 2,37,
       LINCLUDE = 2,79,
       X0 = 0.0,
       Y0 = 0.0,
       Z0 = 0.0,
                                  $END

 $MESH NAME = 'WING',
       LINK = 'PYLON','MISSILE',
       JINCLUDE = 1,80,
       KINCLUDE = 2,40,
       LINCLUDE = 2,29,
       X0 = 0.0,
       Y0 = 0.0,
       Z0 = 0.0,
                                  $END

C_____________________________________________________________________
C                         HOLE DEFINITIONS
C
C  These are the hole boundaries which identify grid points to be
C  eliminated from the computed solution. The location of the boundary
C  ensures sufficient overlap for a valid interpolation stencil.
C_____________________________________________________________________
```



```
C FOR THE RECORD :

C Increasing the boundary minimum effects significantly on the ORPHAN POINTS
C for example, from "thickness" of 5 to 6 points, the number decreases from
C 175 to 18 ORPHAN POINTS only.
C
C
C    HOLE IN MISSILE, WING, OUTBOX MADE BY PYLON
C
C

 $BOUNDARY  NAME = 'HOLE MADE BY PYLON',
                ISPARTOF = 'PYLON',
                MHOLEIN  = 'MISSILE','WING',
                           $END

 $SURFACE ISPARTOF = 'HOLE MADE BY PYLON',

          JRANGE = 8,55,
          KRANGE = 8,21,
          LRANGE = 8,8,
          NVOUT = '-L',
          $END


 $SURFACE ISPARTOF = 'HOLE MADE BY PYLON',

          JRANGE = 8,55,
          KRANGE = 8,21,
          LRANGE = 73,73,
          NVOUT  = '+L',
          $END

 $SURFACE ISPARTOF = 'HOLE MADE BY PYLON',


          JRANGE = 8,55,
          KRANGE = 8,8,
          LRANGE = 8,73,
          NVOUT  = '-K'
          $END


 $SURFACE ISPARTOF = 'HOLE MADE BY PYLON',


         JRANGE = 8,55,
         KRANGE = 21,21,
         LRANGE = 8,73,
         NVOUT = '+K',
         $END

 $SURFACE ISPARTOF = 'HOLE MADE BY PYLON',

         JRANGE = 8,8,
         KRANGE = 8,21,
         LRANGE = 8,73,
         NVOUT  = '-J',
         $END

 $SURFACE ISPARTOF = 'HOLE MADE BY PYLON',

         JRANGE = 55,55,
         KRANGE = 8,21,
         LRANGE = 8,73,
         NVOUT = '+J',
         $END

C
C HOLES IN OUTBOX, WING MADE BY MISSILE
C

 $BOUNDARY NAME = 'HOLE IN OUTBOX/WING MADE BY MISSILE',
         ISPARTOF = 'MISSILE',
         MHOLEIN = 'WING','PYLON',
         $END
```

```
 $SURFACE ISPARTOF = 'HOLE IN OUTBOX/WING MADE BY MISSILE',

         JRANGE = 6,70,
         KRANGE = 9,31,
         LRANGE = 75,75,
         NVOUT = '+L',
         $END

 $SURFACE ISPARTOF = 'HOLE IN OUTBOX/WING MADE BY MISSILE',

         JRANGE = 6,6,
         KRANGE = 9,31,
         LRANGE = 7,75,
         NVOUT = '+J'
         $END

 $SURFACE ISPARTOF = 'HOLE IN OUTBOX/WING MADE BY MISSILE',

         JRANGE = 70,70,
         KRANGE = 9,31,
         LRANGE = 7,75,
         NVOUT = '-J',
         $END

 $SURFACE ISPARTOF = 'HOLE IN OUTBOX/WING MADE BY MISSILE',

         JRANGE = 6,70,
         KRANGE = 9,9,
         LRANGE = 7,75,
         NVOUT = '+K',
         $END

 $SURFACE ISPARTOF = 'HOLE IN OUTBOX/WING MADE BY MISSILE',

         JRANGE = 6,70,
         KRANGE = 31,31,
         LRANGE = 7,75,
         NVOUT = '-K',
         $END


C
C    HOLE IN OUTBOX/MISSILE/PYLON MADE BY WING
C

 $BOUNDARY NAME = 'HOLE MADE BY WING',
         ISPARTOF = 'WING',
         MHOLEIN = 'MISSILE','PYLON',
         $END

 $SURFACE ISPARTOF = 'HOLE MADE BY WING',

         JRANGE = 6,75,
         KRANGE = 9,31,
         LRANGE = 25,25,
         NVOUT = '+L',
         $END

 $SURFACE ISPARTOF = 'HOLE MADE BY WING',

         JRANGE = 6,6,
         KRANGE = 9,31,
         LRANGE = 7,25,
         NVOUT = '+J'
         $END

 $SURFACE ISPARTOF = 'HOLE MADE BY WING',

         JRANGE = 75,75,
         KRANGE = 9,31,
         LRANGE = 7,25,
         NVOUT = '-J',
         $END

 $SURFACE ISPARTOF = 'HOLE MADE BY WING',
```



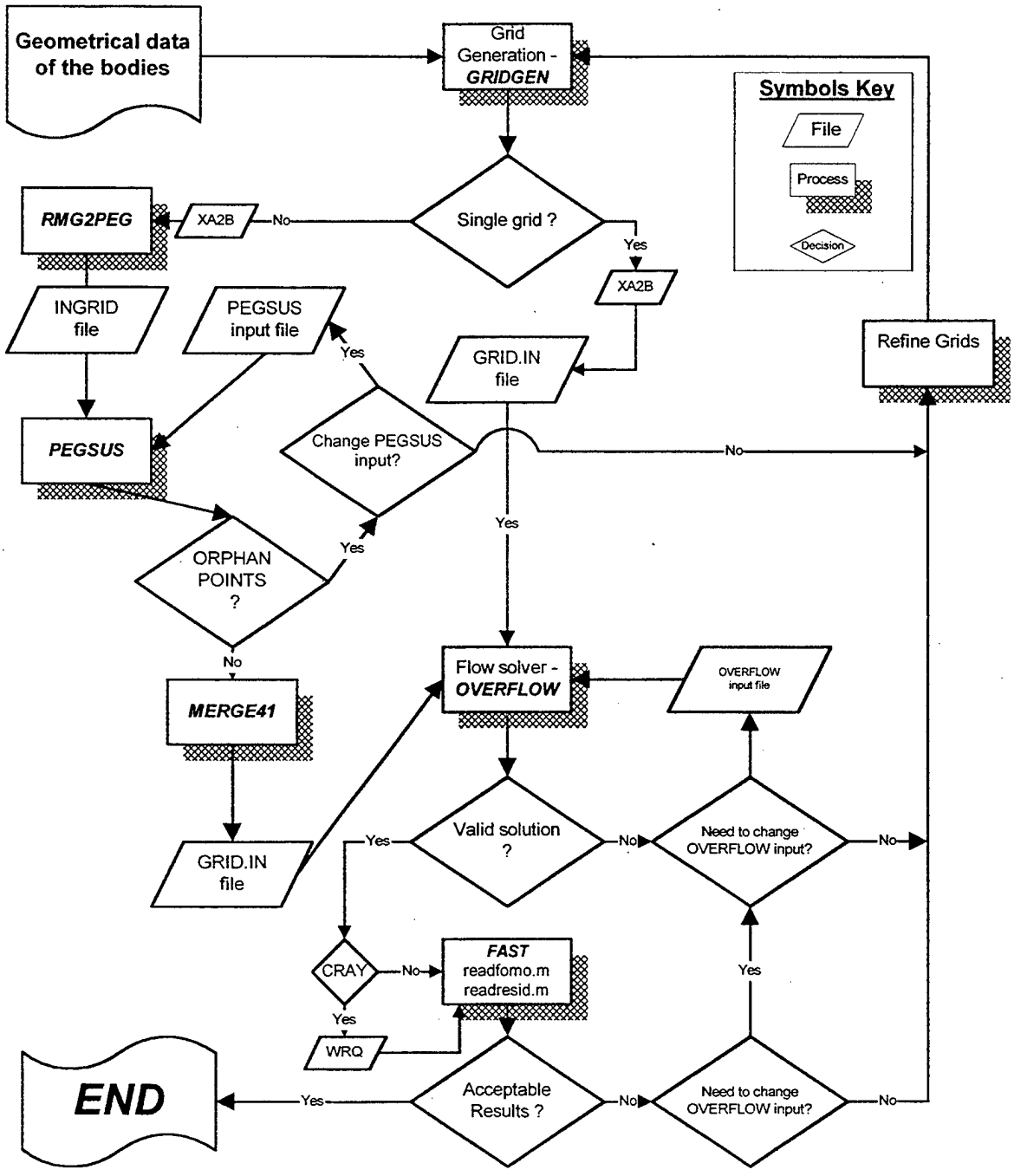
```
         JRANGE = 6,75,
         KRANGE = 9,9,
         LRANGE = 7,25,
         NVOUT = '+K',
         $END

 $SURFACE ISPARTOF = 'HOLE MADE BY WING',

         JRANGE = 6,75,
         KRANGE = 31,31,
         LRANGE = 7,25,
         NVOUT = '-K',
         $END



C_____________________________________________________________
C        BOUNDARY DEFINITIONS
C
C  These are the outer boundary definitions through which other meshes
C  receive information.
C_____________________________________________________________

 $BOUNDARY  NAME = 'OUTER BOUNDARY OF PYLON',
              ISPARTOF = 'PYLON',
              $END

 $SURFACE ISPARTOF = 'OUTER BOUNDARY OF PYLON',
              JRANGE = 1,60,
              KRANGE = 1,28,
              LRANGE = 1,1,
              $END

 $SURFACE ISPARTOF = 'OUTER BOUNDARY OF PYLON',
              JRANGE = 1,60,
              KRANGE = 1,28,
              LRANGE = 80,80,
              $END

 $SURFACE ISPARTOF = 'OUTER BOUNDARY OF PYLON',
              JRANGE = 1,60,
              KRANGE = 1,1,
              LRANGE = 1,80,
              $END

 $SURFACE ISPARTOF = 'OUTER BOUNDARY OF PYLON',
              JRANGE = 1,60,
              KRANGE = 28,28,
              LRANGE = 1,80,
              $END

 $SURFACE ISPARTOF = 'OUTER BOUNDARY OF PYLON',
              JRANGE = 1,1,
              KRANGE = 28,28,
              LRANGE = 1,80,
              $END

 $SURFACE ISPARTOF = 'OUTER BOUNDARY OF PYLON',
              JRANGE = 60,60,
              KRANGE = 28,28,
              LRANGE = 1,80,
              $END


 $BOUNDARY  NAME = 'OUTER BOUNDARY OF MISSILE',
              ISPARTOF = 'MISSILE',
              $END

 $SURFACE ISPARTOF = 'OUTER BOUNDARY OF MISSILE',
             JRANGE = 1,75,
             KRANGE = 1,39,
             LRANGE = 1,1,
             $END
```

```
$SURFACE ISPARTOF = 'OUTER BOUNDARY OF MISSILE',
                JRANGE = 1,75,
                KRANGE = 1,39,
                LRANGE = 80,80,
                $END

$SURFACE ISPARTOF = 'OUTER BOUNDARY OF MISSILE',
                JRANGE = 1,75,
                KRANGE = 1,1,
                LRANGE = 1,80,
                $END

$SURFACE ISPARTOF = 'OUTER BOUNDARY OF MISSILE',
                JRANGE = 1,75,
                KRANGE = 39,39,
                LRANGE = 1,80,
                $END

$SURFACE ISPARTOF = 'OUTER BOUNDARY OF MISSILE',
                JRANGE = 1,1,
                KRANGE = 39,39,
                LRANGE = 1,80,
                $END

$SURFACE ISPARTOF = 'OUTER BOUNDARY OF MISSILE',
                JRANGE = 75,75,
                KRANGE = 39,39,
                LRANGE = 1,80,
                $END



$BOUNDARY  NAME = 'OUTER BOUNDARY OF WING',
                ISPARTOF = 'WING',
                $END

$SURFACE ISPARTOF = 'OUTER BOUNDARY OF WING',
                JRANGE = 1,80,
                KRANGE = 1,39,
                LRANGE = 1,1,
                $END
$SURFACE ISPARTOF = 'OUTER BOUNDARY OF WING',
                JRANGE = 1,80,
                KRANGE = 1,39,
                LRANGE = 30,30,
                $END
$SURFACE ISPARTOF = 'OUTER BOUNDARY OF WING',
                JRANGE = 1,80,
                KRANGE = 1,1,
                LRANGE = 1,30,
                $END
$SURFACE ISPARTOF = 'OUTER BOUNDARY OF WING',

                JRANGE = 1,80,
                KRANGE = 39,39,
                LRANGE = 1,30,
                $END
$SURFACE ISPARTOF = 'OUTER BOUNDARY OF WING',
                JRANGE = 1,1,
                KRANGE = 39,39,
                LRANGE = 1,30,
                $END
$SURFACE ISPARTOF = 'OUTER BOUNDARY OF WING',
                JRANGE = 80,80,
                KRANGE = 39,39,
                LRANGE = 1,30,
                $END
```

```
Job Accounting - Summary Report
*******************************

Job Accounting File Name          : /tmp/nqs.+++++01KS/.jacct15196
Operating System                  : jedi jedi 9.0.2.3 mag.1 CRAY J90
User Name (ID)                    : bpomeran (24268)
Group Name (ID)                   : user (20)
Account Name (ID)                 : A-cr533 (533)
Job Name (ID)                     : multib_run4.nqs (15196)
Report Starts                     : 02/12/97 12:38:18
Report Ends                       : 02/13/97 06:21:36
Elapsed Time                      :        63798      Seconds
User CPU Time                     :      21085.5069   Seconds
Multitasking Breakdown

(Concurrent CPUs * Connect seconds = CPU seconds)
 ---------------   ---------------   -----------

               1 *      11714.5600 =  11714.5600
               2 *       3025.2800 =   6050.5600
               3 *        808.3200 =   2424.9600
               4 *        183.0400 =    732.1600

(Concurrent CPUs * Connect seconds = CPU seconds)
      (Avg.)           (total)          (total)
 ---------------    --------------    -----------

            1.33 *      15731.2000 =  20922.2400

System CPU Time                   :       1197.1922 Seconds
I/O Wait Time (Locked)            :       3898.8845 Seconds
I/O Wait Time (Unlocked)          :        248.7446 Seconds
CPU Time Memory Integral          :     152556.8825 Mword-seconds
SSN Time Memory Integral          :          0.0000 Mword-seconds
I/O Wait Time Memory Integral     :      34630.9733 Mword-seconds
Data Transferred                  :      11367.2868 MWords
Maximum memory used               :          8.9688 MWords
Logical I/O Requests              :     490140
Physical I/O Requests             :     618679
Number of Commands                :          9
Billing Units                     :          0.0000
```

APPENDIX C. CFD PROCEDURE FLOW CHART

(Based on the flow chart in Ref. 7)



APPENDIX D. SCRIPT FILES

A. GENERAL

This appendix includes a sample of script files created during this research to make the CFD process more efficient and "user friendly". The script files are important because of the various files and processes that must be run many times in the same manner. Using a script file assures that the process will be done in the right order.

B. THE FILES

1. script1. A script file used to run RMG2PEG, PEGSUS, MERGE41 and OVERFLOW. Originally the file was created by Buning and can be found in the sample cases of OVERFLOW. It was also used by Barton [Ref. 7].

In this research the file was modified to handle more codes and, hence make the process more efficient.

2. results. A script file created to run the Matlab codes READFOMO.M, READRESID.M and to plot the Cd and residuals.

```
#! /bin/csh -f
#
#    Boaz Pomerantz Feb, 1997
#
#    Script to run PEGSUS, RMG2PEG, MERGE41 and OVERFLOW.
#
#    RUN RMG2PEG
#
/dl/bpomeran/multib/bin/rmg2peg << EOF
3
unformatted
pylonuf.grd
PYLON
missileuf.grd
MISSILE
winguf.grd
WING
INGRID
EOF

#    Run PEGSUS41.
#
#time pegsus41 < peg_1.inp > peg_1.out
pegsus41 < peg_1.inp > peg_1.out

#    RUN MERGE41
/dl/bpomeran/multib/bin/merge41 << EOF
grid.in
EOF

#    Run overflow
#
overflow <overflow.1.inp> overflow.1.out
EOF

#    Clean up
#

%
/bin/rm -f COMPIN INTIN MAPS ORPHAN RSTIN RSTOUT TEMP99
```

```
#   Script file to plot Cd pressure and Cd friction results as well as the
#   residuals. The file also gives a command to print the OVERFLOW input file
#   and the Cray run information (ja_sum.out) for further analysis.

matlab <<EOF

readfomo
1
3
2
2
quit
EOF

matlab <<EOF

readresid
1
3
8
quit
EOF

hp3si -text overflow.1.in
hp3si -text ja_sum.out
```

APPENDIX E. OVERFLOW USER'S INTERFACE IMPROVEMENT SUGGESTIONS

A. GENERAL

OVERFLOW is a code that is still under development. Few recommendations are given to the user in advance to improve the "user friendly" level of the code especially concerning error warnings. If these modifications can be made to the code, future users can save a significant amount of time and frustration in their research.

B. RECOMENDATION FOR IMPROVEMENTS.

1. Symmetric Plane Boundary Conditions.

BCs number 11, 12 and 13 imply symmetry BCs for the X,Y and Z directions respectively [Table 3.3, Ref. 10]. To use these conditions one must add an extra plane so that the symmetry plane will be in between two grid domains and its information will be calculated from these two domains. This action, as detailed earlier is done by GRIDED. However, if this extra plane is not added, a valid ("pseudo") solution is given, which is obviously not physical.

An error message should be given in the OVERFLOW output file if the code recognizes a definition of the symmetry plane without adding an extra plane.

2. Distinguishing Between Equal Grid Size Grids.

Two grids of the same size can provide the same solution "q.save" file, no matter what their shape. The code has to recognize a valid grid and solution pair and prevent mixing a grid and a solution from another grid.

This improvement is important when dealing with a significant number of grids that are only slightly different one from the other (e.g., in the number of Elliptical Solver iterations implied on the grid).

3. Mixing Between NS And Euler Solutions.

The difference between a NS and Euler solution is physically dramatic. However, to flip between the two in the OVERFLOW input file is a very simple command. One can easily change the input file to an NS or Euler solution by changing the VISCJ, VISCK and VISCL parameters in the VISINP group.

To get a valid solution, the BCs must be changed as well (e.g., for a wall, from an inviscid adiabatic wall to a viscous adiabatic wall). If not - a solution will be resolved, but it will not be a physical one; hence, an error message should be raised if there is a mismatch between the VISINP data and the BCs type.

APPENDIX F. MATLAB CODE GENERATES AREA CROSS SECTION DISTRIBUTION

```
%    APPENDIX F - MATLAB CODE GENERATES AREA CROSS SECTION DISTRIBUTION
%
%_____________________________________________________________


disp('***************************************************************')

disp('This code evaluate the cross section area distribution using the grid')
disp('data of the investigated domain.')^M
disp('The code can be used only for domains of which the vertical grid lines')
disp('(constant I)  have a constant "x" value')
disp('***************************************************************')


clear

% Load the domain after the size of the domain was erased from the
% top of the file and the file was "aqaured". i.e. - if the last line
% was not full, "artificial" zeros were added to make the file to b ein
% a form of a matrix.

load jsow_dom.dat;
load pylon_do.grd;
pylon_dom=pylon_do;

domain_m=jsow_dom;

domain_p=pylon_dom;

I= input ('Enter the I grid numbers [along the body]   ')

J= input ('Enter the J grid numbers [across the body]  ')

% Converting the matrix into a one vector "v":
%---------------------------------------------

Size=size(domain_m);

k=0;

for i=1:Size(1);
for j=1:4
k=k+1;
v_m(k)=domain_m(i,j);
v_p(k)=domain_p(i,j);
end
end

for ii=1:I;
xm(ii)=v_m(1+(ii-1)*J);
xp_reg(ii)=v_p(ii);
end

for iiii=1:75;

for iii=1:39
xp(iii)=v_p(iiii+(iii-1)*75);
end

xp_av(iiii)=sum(xp)/39;

end

% From v(2926) (=39*75+1) and on - Y cordinates;

for jj=1:I
for kk=1:J

ym(jj,kk)=v_m(I*J  +J*(jj-1)+kk);
zm(jj,kk)=v_m(2*I*J+J*(jj-1)+kk);

yp(jj,kk)=v_p(J*I  +jj+75*(kk-1));
zp(jj,kk)=v_p(2*I*J+jj+75*(kk-1));

end
end
```



```
% Plotting of the cross section contours from nose to tail^M
%-----------------------------------------------------------

for l=1:39
y10(l)=ym(10,l);
z10(l)=zm(10,l);
y20(l)=ym(20,l);
z20(l)=zm(20,l);
y30(l)=ym(30,l);
z30(l)=zm(30,l);
y40(l)=ym(40,l);
z40(l)=zm(40,l);
y50(l)=ym(50,l);
z50(l)=zm(50,l);
y60(l)=ym(60,l);
z60(l)=zm(60,l);
y70(l)=ym(70,l);
z70(l)=zm(70,l);
end

%plot (y10,z10,y20,z20,y30,z30,y40,z40,y50,z50,y60,z60,y70,z70)
%title ('Cross section area for the CATM nose to tail view')

% Cross section area computation:
%--------------------------------------------------------------

for jj=1:I
for ii=1:J-1
sub_area_m(ii)=abs(zm(jj,ii+1)-zm(jj,ii)) * (-1)*(ym(jj,ii+1)+ym(jj,ii))/2;
sub_area_p(ii)=abs(zp(jj,ii+1)-zp(jj,ii)) * (-1)*(yp(jj,ii+1)+yp(jj,ii))/2;
end

Cross_area_m(jj)=2*sum(sub_area_m);
Cross_area_p(jj)=2*sum(sub_area_p);

end

Cross_area_missile=Cross_area_m;
x_missile=xm;

Cross_area_pylon=Cross_area_p;
x_pylon=xp_av;

% Polynom curve for the missile.
%-------------------------------------------------------------

p3=polyfit(x_missile,Cross_area_missile,3);
poly3=p3(1)*x_missile.^3+p3(2)*x_missile.^2+p3(3)*x_missile+p3(4);

x_missile_shift_pos20=x_missile+20;
x_pylon_shift_pos20=x_pylon+20;
p3_pos20=polyfit(x_missile_shift_pos20,Cross_area_missile,3);
poly3_shift_m_pos20=p3_pos20(1)*x_missile_shift_pos20.^3+p3_pos20(2)*x_missile_shift
_pos20.^2+p3_pos20(3)*x_missile_shift_pos20+p3_pos20(4);^M
poly3_shift_p_pos20=p3_pos20(1)*x_pylon_shift_pos20.^3+p3_pos20(2)*x_pylon_shift_pos
20.^2+p3_pos20(3)*x_pylon_shift_pos20+p3_pos20(4);^M

x_missile_shift_neg20=x_missile-20;
x_pylon_shift_neg20=x_pylon-20;
p3_neg20=polyfit(x_missile_shift_neg20,Cross_area_missile,3);
poly3_shift_neg20=p3_neg20(1)*x_missile_shift_neg20.^3+p3_neg20(2)*x_missile_shift_n
eg20.^2+p3_neg20(3)*x_missile_shift_neg20+p3_neg20(4);
poly3_shift_p_neg20=p3_neg20(1)*x_pylon_shift_neg20.^3+p3_neg20(2)*x_pylon_shift_neg
20.^2+p3_neg20(3)*x_pylon_shift_neg20+p3_neg20(4);

poly3_x_pylon=p3(1)*x_pylon.^3+p3(2)*x_pylon.^2+p3(3)*x_pylon+p3(4);

xe=-75:5:85;
ye=[0 0 0 1 22 62 120 163 175 205 228 245 248 249 249 251 256 266 265 257 221 195 18
2 165 143 126 109 98 91 72 42 39 0];

% xe_min20 - missile forward
% xe_plus20 - missile backward

xe_min20=-80:5:100;
```

```
ye_min20=[1 22 62 120 163 175 187 204 216 216 234 240 245 248 249 239 221 208 204 20
2 196 156 128 110 99 92 77 48 39 39 34 33 39 0 0 0 0];

xe_plus20=-80:5:100;
ye_plus20=[0 0 0 0 0 0 0 0 1 22 80 144 192 207 220 237 251 256 266 276 287 269 257 2
46 226 208 193 181 164 142 121 103 98 52 38 9 0];

xe_min10=-80:5:100;
ye_min10=[0 0 1 22 62 120 163 175 187 204 234 240 245 248 249 249 251 246 236 228 22
5 195 166 143 127 110 98 91 77 48 34 33 39 0 0 0 0];

xe_min5=-80:5:100;
ye_min5=[0 0 0 1 22 62 120 163 175 187 222 240 245 248 249 249 251 256 256 246 239 2
07 183 165 143 127 109 98 91 77 43 33 39 0 0 0 0];

xe_min7p5=[-77.5 -72.5 -67.5 -62.5 -57.5 -52.5 -47.5 -42.5 -37.5 -32.5 -27.5 -22.5 -
17.5 -12.5 -7.5 -2.5 2.5 7.5 12.5 17.5 22.5 27.5 32.5 37.5 42.5 47.5 52.5 57.5 62.5 6
7.5 72.5 77.5 80 82.5 87.5];
ye_min7p5=[0 0 1 22 62 120 163 175 187 211 237 243 247 248.5 249 249.5 252 251 241 2
33 214 188 165 143 126.5 109.5 98 91 77 44 32 36 40 25 0];

xm1=[56.77 56.31 55.04 53.77 51.77 50.77 49.77 46.27 42.77 39.27 35.77 33.55 31.33 2
9.11 26.88 24.66 22.44 20.22 17.99 15.71 14.05 12.33 10.61 8.88 7.16 5.44 3.72 1.99 0
.27 -1.77 -3.83 -5.88 -7.94 -10.00 -12.36 -14.73 -17.10 -19.47 -21.84 -24.21 -26.57 -
28.94 -31.31 -33.68 -36.05 -38.42 -40.78 -43.01 -45.23 -47.45 -49.67 -51.90 -54.12 -5
6.34 -58.56 -60.78 -61.46 -62.13 -62.80 -63.47 -64.13 -64.80 -65.46 -66.12 -66.78 -68
.39 -69.98 -71.56 -73.14 -74.69 -76.24 -77.77 -79.28 -80.78];
Cross_area_m1=[0 1.05 8.02 21.36 25.95 30.95 36.38 42.24 48.83 55.89 63.43 71.45 77.
95 84.73 91.79 99.12 106.73 114.62 122.79 131.23 139.95 144.18 148.47 152.80 157.18 1
61.61 166.09 170.62 175.20 185.00 194.36 203.97 208.15 212.37 216.63 216.63 216.63 21
6.63 216.63 216.63 216.63 216.63 216.63 216.63 216.63 216.63 216.63 216.63 210.17 203
.81 197.54 191.38 185.31 179.33 173.46 167.69 162.01 156.17 150.10 143.81 137.32 130.
67 123.87 116.96 109.96 102.91 87.10 71.21 55.69 41.05 27.81 16.52 7.73 2.03];


% Slope evaluation ,for the various combination of CATM location using
% regression calculation - polyfit of order 1

% -20" case:

xe_min20_s=-5:5:100;
ye_min20_s=[239 221 208 204 202 196 156 128 110 99 92 77 48 39 39 34 33 39 0 0 0 0];

p_min20=polyfit(xe_min20_s,ye_min20_s,1);

% -10" case:

xe_min10_s=0:5:100;
ye_min10_s=[251 246 236 228 225 195 166 143 127 110 98 91 77 48 34 33 39 0 0 0 0];

p_min10=polyfit(xe_min10_s,ye_min10_s,1);

% -7.5" case:

xe_min7p5_s=[2.5 7.5 12.5 17.5 22.5 27.5 32.5 37.5 42.5 47.5 52.5 57.5 62.5 67.5 72.
5 77.5 80 82.5 87.5];
ye_min7p5_s=[ 252 251 241 233 214 188 165 143 126.5 109.5 98 91 77 44 32 36 40 25 0]
;

p_min7p5=polyfit(xe_min7p5_s,ye_min7p5_s,1);


% -5" case:

xe_min5_s=10:5:100;
ye_min5_s=[256 246 239 207 183 165 143 127 109 98 91 77 43 33 39 0 0 0 0];

p_min5=polyfit(xe_min5_s,ye_min5_s,1);

% Aligned case:

xe_s=10:5:85;
ye_s=[266 265 257 221 195 182 165 143 126 109 98 91 72 42 39 0];

p_aligned=polyfit(xe_s,ye_s,1);
```



```
% +20" case:

xe_plus20_s=20:5:100;
ye_plus20_s=[287 269 257 246 226 208 193 181 164 142 121 103 98 52 38 9 0];

p_plus20=polyfit(xe_plus20_s,ye_plus20_s,1);

% CATM alone case;

p_CATM=polyfit(xm1(1:35),Cross_area_m1(1:35),1);

figure (1)

Cd_free=[0.001644,0.001644,0.001644,0.001644,0.001644,0.001644];

shift=[-20,-10,-7.5,-5,0,20];

Cd=[0.003766,0.004629,0.004825,0.004979,0.005195,0.005467];


slope=[p_min20(1),p_min10(1),p_min7p5(1),p_min5(1),p_aligned(1),p_plus20(1)];

slope1=slope(1:6);
Cd1=Cd(1:6);

p=polyfit(abs(slope1),Cd1,2);

slope1=abs(slope1);
slope=abs(slope);

poly_cd=p(1)*slope.^2+p(2)*slope+p(3);

plot(slope,Cd,'o',slope,poly_cd,slope,Cd_free);
axis([2.4 3.7 0 0.006])
title('Cd as a function of the slope of the afterbody area distribution')
xlabel ('Slope of the afterbody area distribution')
ylabel ('Cd')
grid
gtext ('shift=-20[in]')
gtext ('shift=-10[in]')
gtext ('shift=-7.5[in]')
gtext ('shift=-5[in]')
gtext ('Aligned')
gtext ('shift=+20[in]')
gtext ('Free flight condition - without the pylon')


figure (2)

% 4 Plots configuration:

subplot (2,2,1);

ye1=p_aligned(1)*xe_s+p_aligned(2);

plot (xe,ye,xe_s,ye1)

axis([-100 100 0 300])
title ('Aligned case')
grid

subplot (2,2,2);

ye_min20_1=p_min20(1)*xe_min20_s+p_min20(2);

plot (xe_min20,ye_min20,xe_min20_s,ye_min20_1);

axis([-100 100 0 300])
title ('Forward case -20"')
grid

subplot (2,2,3);

ye_plus20_1=p_plus20(1)*xe_plus20_s+p_plus20(2);

plot (xe_plus20,ye_plus20,xe_plus20_s,ye_plus20_1)
```

```
axis([-100 100 0 300])
title ('Backward case +20"')
grid

subplot (2,2,4);

Cross_area_ml_s=p_CATM(1)*(xml(1:35))+p_CATM(2);

plot (xml+20,Cross_area_ml,xml(1:35)+20,Cross_area_ml_s)

axis([-100 100 0 300])
title ('Missile without the Pylon')
grid
```

APPENDIX G. FEW RECOMMENDATIONS FOR THE CRAY USERS.

Most of the jobs, especially the heavy multiblock cases, were run on the Cray computer in the Computer Center of NPS. The interface format of the user with the Cray is by job submitting.

The current Cray model is Y-MPEL89, and consists of 8 CPU's sharing 256 Mwords of memory.

As for this research, the Cray (Jedi) was able to deal with only ten jobs simultaneously. In any case, a user cannot run more then one job at a time. The jobs are sent by ".nqs" files [App. H], which define the job type, commands, and priority. The only approved priority that can be used by the students are the "reg" or "econ" types. The "prem" type is forbidden to be used. Because of the this, jobs can wait for a long time to be executed after being submitted by the user.

To avoid too much waiting time when the user identifies a "busy" Cray (by typing the "nqstat" command in the Cray), the runs can be divided into relatively low numbers of iterations, and the "restart" feature of OVERFLOW can be used to define low amount of memory required for the each job.

The job's priority decision is done, not according to "first submitted, first executed" but, according to the amount of time and memory required as defined in the nqs file. Long, "heavy" jobs wait in the queue a longer time.

A practical method to have an idea about the size of a job, how much time it runs, how much CPU it consumes and more is through "ja_sum.out" file. This file contains some useful information that might help one to plan his jobs. An example is given in App. B and the command lines to get the file are given within the "nqs" file in App. H.

APPENDIX H. NQS FILE SAMPLE

A. GENERAL

Since the interaction with the Cray is through batch files and not an "on line" process as is usually performed, an "nqs" file has to be created, in which the procedure of the commands is given.

The benefits of the nqs file were not fully used in this research. This file can be used to create a complicated procedure, including running jobs, transferring files, and changing directories.

```
#QSUB -q reg -lT 25200 -lt 25200 -lM 10Mw
cd /dl/bpomeran/multib/runs/run4
ja
make_rmg_peg
ja -st > ja_sum.out
```

APPENDIX I. LOG OF THE MAIN CASES THAT WERE INVESTIGATED

COMMENTS

1. Angle of attack

Because of the grid directions, ALFA, as defined in the OVEFLOW input file, is actually BETA and BETA is negative ALFA. The values detailed in Table 1 are as defined in OVERFLOW input file.

2. PEGSUS input files

The input file for PEGSUS was modified by defining the missile and the pylon as making holes in each other (the original file specified that the pylon makes a hole in the missile, and not vice versa; however it was found that the results are pretty much the same.

3. Viscous / Inviscid cases

Unless mentioned otherwise, the run is inviscid (Euler solution). If an NS solution is obtained, the directions of viscosity are mentioned as well (J,K,L).

4. CRAY / SGI runs

Runs defined as RUN* are CRAY cases, usually multiblock runs. Runs defined as "garth*" are SGI cases, and are usually simple, single block cases.

5. Blocks shifting for Area Rule investigation

In the case of a block that was shifted backward or forward, the magnitude of shifting in [in] is defined in parenthesis adjacent the block (e.g., M(+20")).

6. Reynolds number and Temperature at infinity values

Most of the cases (unless written otherwise), the Re number and T infinity were Re=0.166667E6 and TINF=576 [deg. Renkin] respectively. The cases investigated specifically in this research were run with the more realistic data of Re=8.713E6 and TINF=518.42 [deg. Renkin], in those cases, it is mentioned specifically in the following table.

Table I1. Log of the main cases that were investigated

Case #	RUN #	Configuration	Time interval - DT [sec]	CFL	# of iterations	Mach #	Angle Of Attack - α	Slide Angle - β	Cd Pylon	Cd Missile	Cd Wing	Residuals	Remarks
1	RUN1	Pylon+ Missile Aligned	0.01	3	2000	0.8	0	0	0.002058	0.000227			
2	RUN2	Pylon+ Missile(+20")	0.01	3	1000	0.85	0	0	0.00759	0.00547			
3	RUN3	Pylon+ Missile(-20")	0.01	3	1000	0.85	0	0	0.00.759	0.00376			
4	RUN1	Pylon+ Missile Aligned	0.01	3	1000	0.85	0	0	0.00759	0.00519			
5	RUN23	Missile	0.01	3	2000	0.8	0	0	-	0.002			
6	RUN25	Missile	0.05	3	1000	0.8	0	0	-	0.0038			
7	RUN30	Pylon	0.001	5	3000	0.8	0	0	0.00759				
8	garth1	Missile	0.01	5	3000	0.8	0	0	0.00759	0.00500			
9	garth2	Missile	0.01	3	3000	0.8	0	0	-	0.002			
10	garth1	Missile	0.01	5	1000	0.85	0	0	-	0.001644			
11	garth2	Missile	0.01	5	1000	0.85	0	0	-	-			
12	RUN4	P+M+W	0.01	3	1000	0.85	0	0	0.00759	0.004092	0.00227		
13	RUN1	P+M	0.01	3	1000	0.6	0	0	0.008569	0.003871			
14	RUN2	P+M(+20")	0.01	3	1000	0.6	0	0	0.008569	0.004149			
15	RUN3	P+M(-20")	0.01	3	1000	0.6	0	0	0.008569	0.003045			
16	garth1	Missile	0.01	5	2000	0.6	0	0		0.001188			
17	garht2	Missile	0.01	5	1000	0.6	3	0		-0.1266			Negative Cd due to α/β exchange

18	garth1	Missile	0.01	5	1000	0.85	5	0		-0.102			Negative Cd due to α/β exchange
19	garth2	Missile	0.01	5	1000								
20	garth1	Missile	0.01	5	1000	0.95	0	0		0.002679			
21	garth2	Missile	0.01	5	1000	0.95	3	0		-0.047			Negative Cd due to α/β exchange
22	garth1	Missile	0.01	5	1000	1.2	0	0		0.010751			
23	garth2	Missile	0.01	5	1000	0.95	-3	0		0.005523			
24	garth1	Missile	0.01	5	1000	0.85	0	3		0.001654			Cases to check the definition and direction of α,β.
25	garth2	Missile	0.01	5	1000	0.85	0	-3		0.001725			
26	garth3	Missile	0.01	5	1000	0.85	3	0		0.001654			
27	garth4	Missile	0.01	5	1000	0.85	-3	0		0.001725			
28	RUN1	P+M	0.01	5	1000	0.85	0	0	0.00759	0.005195			From now on, the pegsus input file is modified
29	RUN2	P+M(+20")	0.01	5	1000	0.85	0	0	0.00759	0.005467			
30	RUN3	P+M(-20")	0.01	5	1000	0.85	0	0	0.00759	0.003766			
31	RUN4	P+M+W	0.01	5	1500	0.85	0	0	0.00759	0.004092	0.002268		
32	RUN5	P+M(+20")+W	0.01	5	1500	0.85	0	0	0.00759	0.004671	0.002308		
33	RUN6	P+M(-20")+W	0.01	5	1500	0.85	0	0	0.00759	0.003309	0.0022		
34	RUN7	P+M(-7.5")+W	0.01	5	1500	0.85	0	0	0.00759	0.003811	0.002246		
35	RUN4	P+M+W	0.01	5	1500	0.95	0	0					
36	RUN5	P+M(+20")+W	0.01	5	1500	0.95	0	0					
37	RUN6	P+M(-20")+W	0.01		1500	0.95	0	0					
38	RUN1	P+M	0.01	5	1500	0.95		0					
39	RUN2	P+M(+20")	0.01	5	1500	0.95	0	0					
40	RUN3	P+M(-20")	0.01		1500	0.95	0	0					
41	RUN8	P+M(-7.5")	0.01	5	1500	0.95	0	0	0.01450	0.006000			
42	RUN8	P+M(-7.5")	0.01	5	1500	0.85	0	0	0.00759	0.004825			

43	garth5	Missile	0.01	5	1000	0.85	0	0		0.001823			Vis on J,K,L
44	garth6	Missile	0.01	5	1000	0.85	0	0		0.001773			L
45	garth1	Missile	0.01	5	1000	0.85	0	0		0.001800			K,L
46	garth2	Missile	0.01	5	1000	0.95	0	0		0.002687			J,K,L
47	garth1	Missile	0.01	5	1000	0.85	0	3		0.001888			J,K,L AOA defined by β and represents negative α
48	garth2	Missile	0.01	5	1000	0.85	0	8		0.002625			J,K,L
49	garth5	Missile	0.01	5	1000	0.98	0	0		0.003285			J,K,L
50	garth6	Missile	0.01	5	1000	1.02	0	0		0.004806			J,K,L
51	garth7	Missile	0.01	5	1000	0.85	0	0		0.001769			J,K,L
52	garth8	Missile	0.01	5	1000	0.6	0	0		0.001289			J,K,L
53	garth1	Missile	0.01	5	1000	0.85	0	15		0.00578			J,K,L
54	garth2	Missile	0.01	5	1000	0.85	0	30		0.03348			J,K,L
55	RUN1	P+M	0.01	3	1000	1.02	0	0	0.00782	0.0100306			
56	RUN2	P+M(+20")	0.01	3	1000	1.02	0	0	0.00782	0.0099247			
57	RUN3	P+M(-20")	0.01	3	1000	1.02	0	0	0.00782	000758			
58	RUN4	P+M+W	0.01	3	1000	1.02	0	0	0.00782	0.009125	0.004853		
59	RUN5	P+M(-20")	0.01	3	1000	1.02	0	0	0.00782	-.009988	0.00497		
60	RUN6	P+M(-20") +W	0.01	3	1000	1.02	0	0	0.00782	0.007408	0.00475		
61	RUN9	P+M(-5")	0.01	3	1000	1.02	0	0	0.00782	0.00306			
62	RUN10	P+M(-10")	0.01	3	1000	1.02	0	0	0.00782	0.01003			
63	RUN11	P(+20")+M	0.01	3	1000	0.85	0	0	0.00759	0.003766			
64	RUN12	P(-20")+M	0.01	3	1000	1.02	0	0	0.00782	0.009924			
65	RUN8	P+M(-7.5)	0.01	3	1000	0.6	0	0	0.008568	0.003550			
66	garht1	Missile	0.01	3	1000	0.85	0	-3		0.0020417			Viscous on J,K,L
67	garh2	Missile	0.01	3	1000	0.85	0	-6		0.002608			J,K,L
68	garh5	Missile	0.01	3	1000	0.85	0	-9		0.00348			J,K,L
69	garht6	Missile	0.01	3	1000	0.85	0	-12		0.004745			J,K,L
70	garht7	Missile	0.01	3	1000	0.95	0	-15		0.006497			J,K,L
71	garht8	Missile	0.01	3	1000	0.85	0	-30		0.0252			J,K,L
72	RUN1	P+M	0.01	3	1000	0.85	0	0	0.00733	0.00504			Viscous on J,K,L
73	RUN2	P+M(+20")	0.01	3	1000	0.85	0	0	0.007757	0.00566			J,K,L

74	garth1	Missile	0.01	5	1000	0.85	0	0		0.001899			Viscous on J,K,L, Re=8.713 E6, TINF=518 .42
75	garth2	Missile	0.01	5	1000	0.85	0	0		0.001645			Inviscid Re=8.713 E6, TINF=518 .42
76	RUN4	M+P+W	0.01	3	1500	0.85	0	0	0.007947	0.0417	0.00430		Viscous on J,K,L, Re=8.713 E6, TINF=518 .42
77	RUN5	M+P+W	0.01	3	1500	0.85	0	0	0.007591	0.00467	0.004756		Inviscid Re=8.713 E6, TINF=518 .42
78	garth5	Missile	0.01	5	1000	0.5	0	0		0.001178			Inviscid
79	garth6	Missile	0.01	5	1000	0.6	0	0		0.001167			Inviscid
80	garth7	Missile	0.01	5	1000	0.7	0	0		0.001233			Inviscid
81	garth8	Missile	0.01	5	1000	0.8	0	0		0.001461			Inviscid
82	garth9	Missile	0.01	5	1000	0.95	0	0		0.002718			Inviscid
83	garth10	Missile	0.01	5	1000	0.98	0	0		0.003459			Inviscid
84	garth11	Missile	0.01	5	1000	1.02	0	0		0.005111			Inviscid
85	garth12	Missile	0.01	5	1000	1.1	0	0		0.010500			Inviscid
86	garth13	Missile	0.01	5	1000	1.2	0	0		0.010770			Inviscid
87	garth5	Missile	0.01	5	1000	1.3	0	0		0.01129			Inviscid
88	garth6	Missile	0.01	5	1000	1.4	0	0		0.01156			Inviscid
89	garth7	Missile	0.01	5	1000	1.5	0	0		0.01146			Inviscid
90	garth8	Missile	0.01	5	1000	1.6	0	0		0.011245			Inviscid
91	garth9	Missile	0.01	5	1000	1.7	0	0		0.01104			Inviscid
92	garth10	Missile	0.01	5	1000	1.8	0	0		0.01082			Inviscid
93	garth1	Missile	0.01	5	1000	0.5	0	0		0.001427			Viscous
94	garth2	Missile	0.01	5	1000	0.6	0	0		0.0014405			Viscous
95	garth3	Missile	0.01	5	1000	0.7	0	0					Viscous
97	garth4	Missile	0.01	5	1000	0.8	0	0		0.001741			Viscous
98	garth5	Missile	0.01	5	1000	0.85	0	0		0.001899			Viscous

99	garth6	Missile	0.01	5	1000	0.98	0	0		0.0033218			Viscous
100	garth7	Missile	0.01	5	1000	1.02	0	0		0.004806			Viscous
101	garth8	Missile	0.01	5	1000	1.1	0	0		0.009755			Viscous
102	garth9	Missile	0.01	5	1000	1.2	0	0		0.010006			Viscous
103	garth10	Missile	0.01	5	1000	1.3	0	0		0.0104604			Viscous
104	garth11	Missile	0.01	5	1000	1.4	0	0		0.0107191			Viscous
105	garth12	Missile	0.01	5	1000	1.5	0	0		q.bomb			Viscous
106	garth13	Missile	0.01	5	1000	1.6	0	0		q.bomb			Viscous
107	garth14	Missile	0.01	5	1000	1.7	0	0		q.bomb			Viscous
108	garth15	Missile	0.01	5	1000	1.8	0	0		q.bomb			Viscous
109	garth3	Missile	0.01	5	1000	1	0	0		0.004152			Inviscid
110	garth4	Missile	0.01	5	1000	1	0	0		0.003907			Viscous

APPENDIX J. A PRINCIPAL PROCESS OF CFD ANALYSIS

A. General

A concise flow chart describing the process of CFD is given in Ref. 7. The chart includes most of the phases in the process; however, it was found that a detailed description of the different stages including some emphasis on particular important points can be very helpful for the user.

Therefore, it is recommended to use the flow chart, given in Appendix C parallel to the literal description.

The description will concentrate on the multiblock case as was done in this research (i.e., the tools that we used, e.g., GRIDGEN and FAST). From this general case, it is obvious how to handle the single block case.

B. The Process

1. Generate the grids (using GRIDGEN or any other grid generation tool).

2. Edit the grids and remove the "1" from the top left corner of the file. This number describes the number of blocks included in each file, and has to be removed in order to run GRIDED and "xa2b" on the file. It is recommended to rename the "1 less" file in order to prevent long editing times in case the need to read the grid in GRIDGEN arises again.

3. Visualize the grids using FAST; write down the directions and orientation of I, J, K in order to plan the changes that have to be done on the grid.

4. Decide on a particular grid direction and orientation that will be consistent through the research. That way, only minor changes will have to be made to the OVERFLOW, PEGSUS input files, and quite a few sources for mistakes will be eliminated. In our case, for example, through all the research, the missile surface was K=1, the front sting was I=1 and the lower symmetry plane was J=1.

5. Run GRIDED on the file and change the orientations and directions of the grid as needed. In addition, add an extra plane in the symmetry planes.

6. Convert the file created by GRIDED to a binary format using "xa2b" code. Pay attention that the file has to be compiled differently if the process will continue to be done on the Cray or locally on the SGI machines.

7. In this stage, it is recommended to run each grid separately on a simple and short (approximately 50 to 100 iterations) OVERFLOW case to check, in principle the convergence of the solution and absence of negative cells. If a negative cell error were found in the OVERFLOW output file, before going back to the long process of grid generation and modification, it is recommended to run GRIDED again on the original file (i.e., do steps "b" and so on again), and/or compile the conversion file "xa2b" and run it again. If this does not help, a modified grid has to be generated with different grid line spacing (distribution) and/or with an elliptical solver implied on the domains. It was found that negative cells appear sometimes without any "physical" reason.

8. Prepare the grids to run PEGSUS by applying the RMG2PEG code on the grids. It is very convenient and saves time if a script file is generated to run the process of RMG2PEG -> PEGSUS -> MERGE41 -> OVERFLOW (Appendix D). Run the script file for a small amount of OVERFLOW iterations to verify the whole process. Check Orphan Points in the PEGSUS output file. It must be mentioned that OPs will not prevent a valid OVERFLOW solution. As mentioned before, OPs is a point that is not being

updated; however, it is recommended to get rid of all the OPs to prevent "holes" in the solution. In the case of OPs, change the PEGSUS input file according to the mechanization's detailed earlier.

9. Once a good PEGSUS input file has been achieved, OVERFLOW can be run for a significant number of iterations. The physical solution can be analyzed starting from residual values of approximately 10E-5 (ΔQ); however, the other way around is not true, these residual values do not imply a good solution. A physical value of aerodynamic coefficients and a reasonable flow pattern has to be found.

10. The solution can now be visualized using FAST, in the case of a Cray run, the "Q" file, "q.save", has to be converted to a SGI format using the "wrq" file, and the "grid.in" file has to be converted to a SGI format using the "wrgr" file. Notice that the file created by "wrgr", "gr.sgi", includes the blanking points as well as the whole other grid points. Therefor, if one wants to view the blanking scheme, that file has to be used in FAST. In the case of a local, SGI run, the solution "q.save" can be read directly as a solution/unformatted file.

11. More OVERFLOW iterations or modifications of the grids can be done on the base of the successful run.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center 8725 John J. Kingman Rd., Ste 0944 Ft. Belvoir, VA 22060-6218	2
2.	Dudley Knox Library Naval Postgraduate School 411 Dyer Rd. Monterey, CA 93943-5101	2
3.	Prof. D. J. Collins, Chairman, Code AA/Co Department of Aeronautics and Astronautics Naval Postgraduate School Monterey, CA 93943-5101	1
4.	Prof. O. Biblarz, Code AA/Bi Department of Aeronautics and Astronautics Naval Postgraduate School Monterey, CA 93943-5101	4
5.	Prof. G. Hobson Code AA/Ag Department of Aeronautics and Astronautics Naval Postgraduate School Monterey, CA 93943-5101	1

6. Prof. G. Lindsey Code AA/Li — 2
Department of Aeronautics and Astronautics
Naval Postgraduate School
Monterey, CA 93943-5101

7. Major Boaz Pomerantz — 3
Israeli Air Force Headquarters
Galil 84, Reut, 71908
Israel